ARIZONA STATE UNIVERSITY

Department of Electrical Engineering
Arizona State University
Tempe, AZ 85287-7206
Tel: (480) 965-5311
Fax: (480) 965-8325
Email: spanias@asu.edu

20 July 2003

2003 Premier Award c/o NEEDS
3115 Etcheverry Hall
University of California at Berkeley
Berkeley, CA 94720-1750

Dear Sir/Madam

I respectfully submit my engineering education software entitled Java-Digital Signal Processing (J-DSP), for consideration for the 2003 Premier award. Enclosed are 15 copies of the J-DSP submission packet.

We refer to J-DSP as "*labware*" because it enables student users to perform online simulations and computer experiments over the Internet. The J-DSP concept is due to Professor Andreas Spanias who is also the chief software designer and the main author of the labware. Collaborators include graduate students who have assisted the main author with various software tasks and colleagues who used the software in their classes.

The J-DSP project is sponsored by Arizona State University and by NSF CCLI EMD 0089075. The copyright owner of the J-DSP source and executable code is the Arizona State Board of Regents that is by law, holder of all intellectual property generated by Arizona State University. It is a non-profit project and therefore J-DSP is freely accessible over the Internet, not only to ASU students, but also to students at other Universities and DSP practitioners. J-DSP executables (ISBN 0-9724984-0-0) have already been freely disseminated to more than thirty institutions worldwide. The dissemination process is still ongoing and therefore we can grant NEEDS the permission to non-exclusively disseminate the J-DSP executable Version 1, Copyright 1997-2003 as packaged on the CD-ROM ISBN 0-9724984-0-0 that accompanies this submission packet.

To evaluate the software please visit http://jdsp.asu.edu and follow the designated link for the NEEDS award. Because J-DSP is not an e-book or a multimedia presentation but a comprehensive online environment we provide a step by step procedure for the reviewers to familiarize themselves and evaluate the labware and all associated materials.

We will be obliged if the evaluation committee evaluates this software on its intended purpose which is to provide a DSP simulation environment and online laboratory experiences for DSP courses. Please note that our labware is not intended to compete with commercial packages such as the world renowned MATLAB/Simulink which we also use and promote in our classes at ASU. Our freely accessible non-for-profit J-DSP software simply provides an environment only for compact educational simulations on any platform that is accessible through any browser such as the MS Explorer. The software was specifically developed for DSP courses.

Thank you for considering our Engineering labware J-DSP.

Sincerely,

Andreas Spanias

Andreas Spanias, Ph.D.
Professor and PI J-DSP project
IEEE Fellow

Java-DSP (J-DSP); An On-line DSP Simulation Tool enabling Web-based Computer Labs

J-DSP Concept and Software by: Prof. Andreas Spanias, Ph.D.
Multidisciplinary Distance Learning Initiative (MIDL)
Department of Electrical Engineering, Arizona State University
Tempe, AZ 85287-5706, Email: spanias@asu.edu

1. Background

Java-DSP (J-DSP) (http://jdsp.asu.edu) is an educational program that enables on-line simulations and web-based computer laboratories for use in Digital Signal Processing (DSP) courses. The initial version of J-DSP has been developed in the ASU MIDL lab and tested in a senior-level Electrical Engineering DSP course (EEE 407) during the academic year 1996-97. Portions of the software have been displayed in the 1997 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP-97) in Munich and a paper on this pilot study was published in the ICASSP-98 proceedings [1]. Since then, J-DSP has been significantly upgraded with new functionality and its associated laboratory exercises are being continuously improved [2-11]. J-DSP is an object-oriented environment that enables students to establish and run educational DSP simulations, such as, digital filter design, signal analysis, speech signal processing, FFT-based spectral analysis, etc. J-DSP provides a user-friendly graphical user interface (GUI) that is freely and universally accessible from the Internet using a browser such as the MS Internet Explorer®[1]. The J-DSP GUI enables visual programming of DSP tasks on any platform directly from the browser. In Appendix A, we show several figures that present the GUI of the software. We also encourage the NEEDS reviewers to visit the web site (http://jdsp.asu.edu) and/or use the enclosed CD ROM and view the AVI files that demonstrate some key functions of J-DSP. J-DSP is freely accessible, platform independent, and created for non-for-profit use. J-DSP enables:

- students to establish and run quick simulations on the web,
- students to see on-line interactive demos embedded in web lectures,
- students to perform on-line computer laboratories on the web,
- instructors to assign J-DSP-based exercises,
- instructors to embed interactive demos in their web lecture content.

The J-DSP Version 1 (CD-ROM ISBN 0-9724984-0-0) is approximately 42,000 lines of Java code. To fully support J-DSP, Prof. Spanias has developed a series of J-DSP laboratory exercises (see Appendix D) that have been used in EEE 407. Assessment of J-DSP is carried on a semester-by-semester basis and web-based assessment instruments have been developed and posted on the J-DSP web site. A more formal and detailed assessment of the J-DSP software and its associated EEE 407 lab exercises was conducted recently and results will be presented at the 2003 IEEE/ASEE Frontiers in Education conference [6] (see copy of the assessment report and relevant paper in Appendix C). Assessment results from this award have recently been submitted to the Online Evaluation Resource Library (OERL). A comprehensive manual for J-DSP was also developed and posted on the internet and a hard copy of this manual is in Appendix F. The free dissemination of J-DSP to other institutions started in the Fall of 2002. A dissemination package was developed (Appendix C) and the J-DSP software and labs have been disseminated to many prominent faculty members in ranked engineering programs. Universities that obtained free the J-DSP Version 1 executable software (ISBN 0-9724984-0-0) included MIT, U. Penn, Georgia Tech., University of Minnesota, Rice University, and several others (see complete list Appendix C-3). In the summer of 2000, NSF awarded a CCLI-EMD grant that enabled us to upgrade the DSP functionality and extend the use of J-DSP in other related courses such as communications, image processing, and controls [11]. The work on the NSF J-DSP project is ongoing and several new functions are being developed. Such functions involve:

[1] MS Internet Explorer is registered by Microsoft Incorporated. MATLAB is registered by The Mathworks. J-DSP is sponsored by ASU and NSF CCLI EMD DUE 0089075.

- *J-DSP Scripts* enabling instructors to effortlessly create J-DSP simulations and seamlessly integrate them (Appendix A – Fig. A.3) in their web content or on-line lectures (see reference [3] and J-DSP manual in Appendix F). This J-DSP feature was developed by creating J-DSP scripts for every function that are embeddable in html content. The task was quite involved and comparable to developing a complete computer interpreter (entirely completed for DSP) [3].
- *Accompanying servlets* to allow students to submit their J-DSP lab reports for EEE 407 on the network [2] (completed).
- *Advanced DSP functionality* for Speech Processing [9], Spectrograms[5] (partially completed).

Other functions being developed in the NSF J-DSP project include: controls [4], image processing [8], and communications [10]. Also as part of the NSF grant we are developing further the J-DSP GUI. In particular, we are developing an interface to MATLAB (see Appendix A - Fig. A.4) and we will develop a new GUI that allows seamless integration of J-DSP functionality with streaming video/audio (Fig. A.5).

1.1 The J-DSP Graphical User Interface

J-DSP has a rich suite of signal processing functions that facilitate interactive on-line simulations of modern statistical signal and spectral analysis algorithms, filter design tools, QMF banks, and state-of-the-art vocoders. All functions in J-DSP appear as graphical blocks that are divided into groups according to their functionality. Selecting and establishing individual blocks can be done using a drag-and-drop-process. Each block is linked to a signal processing function. Figure 1 shows the J-DSP editor environment. By connecting blocks together, a variety of DSP systems can be simulated. Signals at any point of a simulation can be examined and block parameters can be edited through dialog windows.

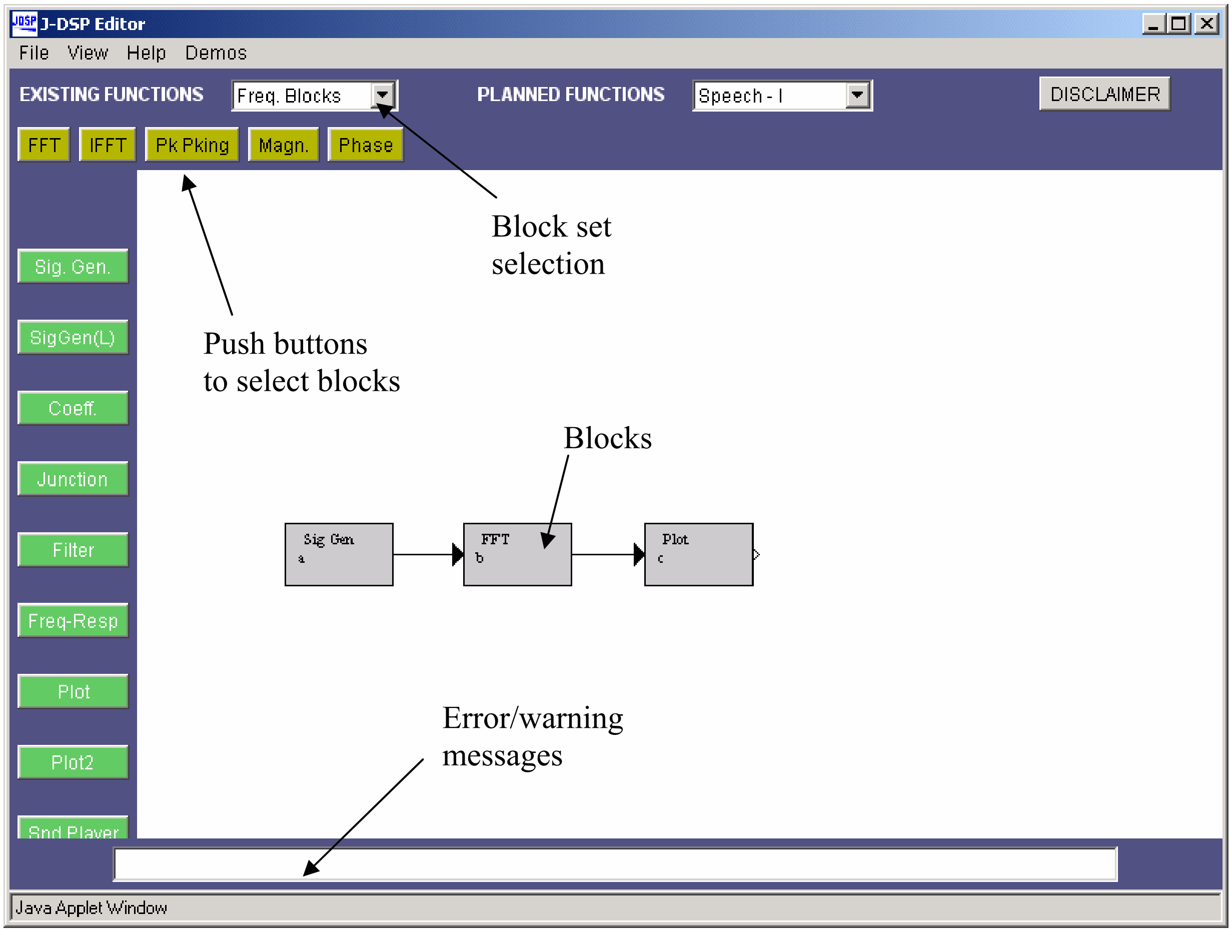


Fig.1. J-DSP Graphical User Interface

System execution is dynamic, which means that any change at any point of a system will automatically take effect in all related blocks. Any number of block windows can be left open to enable viewing results at more than one point in the system. The NEEDS panel reviewer may visit the web site and first view some of the instructional AVI files to become familiar with J-DSP. A step-by-step process has been posted on the web (http://jdsp.asu.edu) to guide the reviewers through J-DSP. Student users are typically assigned a small signal processing task as part of their introductory lab (Appendix D) in EEE 407 and then asked to run specific simulations and make measurements to verify theory using J-DSP. In class evaluations students reported that it typically took them just a few minutes to become familiar with the J-DSP GUI. A J-DSP *Quick Reference / Getting Started Guide* was also disseminated.

2. Target Audiences

- **Principal Audience:** The J-DSP was originally intended to provide on-line computer laboratory experiences to Electrical Engineering students taking introductory DSP classes. These are typically undergraduate students at the junior or senior level. The version of J-DSP that has been completed and submitted for evaluation by the NEEDS committee is the one currently used in the EEE 407 DSP class at Arizona State University. This course is a senior undergraduate elective which is also attended by first semester graduate students usually specializing in DSP or Communications. The J-DSP labs (Appendix D), that are performed using the J-DSP software, comprise the one-credit laboratory portion of the 4-credit EEE 407 course. All the students in that class are required to attempt the J-DSP laboratory exercises, prepare lab reports, and submit them electronically using the tools on the EEE 407 laboratory web site.

- **ASU Colleagues and Students in other Areas:** With the new functionality in communications [8], controls [4], and image processing [10] my colleagues working in these areas have used some of the J-DSP functions and supporting software in assigned homework in their classes. Descriptions of exercises and preliminary assessment results are reported in the corresponding papers [8,4,10].

- **Student Visitors and Practitioners:** Our J-DSP software is open to all students and practitioners worldwide. Our web site is set up to guide them through a J-DSP exercise and then an on-line evaluation and assessment.

- **High School Students:** We have recently developed J-DSP functions to introduce high school students to DSP and Multimedia technologies [7].

- **Colleagues at other Universities:** Formal dissemination to colleagues at other universities started in the Fall of 2002. We have several colleagues that obtained copies of J-DSP (Appendix C-3).

3. Learning Objectives

3.1 General objectives of J-DSP and the associated J-DSP laboratory exercises are:

- To provide on-line laboratory experiences to undergraduate DSP students in Electrical Engineering and enable programmable Internet simulations with embedded animations.
- To accelerate leaning by exposing students to hands-on manipulation of signals and DSP systems.
- To provide DSP problems and design exercises that reinforce the theory learned in class and provide intuition and complementary information usually not available in lectures and text books.

3.2 Specific J-DSP objectives relating to specific DSP topics are:

- to provide hands-on experiences by simulating DSP systems (filters, FFTs, etc),
- to provide visualization examples relating time-domain, z-domain, and Fourier domain,
- to provide hands-on experiences with FIR and IIR digital filter design,
- to familiarize students with the use of windows and FFTs in spectral analysis,
- to expose students to quantization effects via J-DSP on-line simulation,

- to expose students to filter banks of the type used in MP3 and other compression formats,
- to expose students to random signal analysis by simulating filters with random inputs,
- to expose students to periodograms, correlograms, and linear prediction,
- to provide students with an introduction to speech processing.

4. General J-DSP Labware Description and Usage

The software is used to support the laboratory portion of EEE 407. Laboratory exercises (sample copy in Appendix D) are assigned on a weekly basis. These laboratory exercises actually reside on the WWW. While access to the J-DSP program is free and universal, access to the laboratory page is secure because it invokes student information, grades etc. Students perform EEE 407 labs as follows:

- The student follows instructions on the secure EEE407 laboratory web page and starts the J-DSP program by pressing a button that activates the software. Before starting J-DSP students are reminded to download the Java virtual machine (link provided on the web) that enables the MS Internet Explorer browser to run J-DSP.
- The student then proceeds to form and execute the required J-DSP simulations, generate graphs and data, and then fill out an electronic report form. The electronic report allows the students to answer questions, attach J-DSP data and graphs, provide comments and textual interpretation of results, and respond to an electronically graded quiz. This form is processed by servlets (described in reference [2]) which produce an electronic HTML report for each student that compiles the student responses, graphics, etc. The instructor then evaluates and grades the report electronically. An electronic grade book is interfaced with the J-DSP labware. Specifics on the report submission are given in reference [2] and samples of reports are given in Appendix D- 3.

4.1 Description of a Sample J-DSP Laboratory Exercise and the Associated Learning

In J-DSP Lab 2: The z transform and Frequency Responses (Appendix D-1)

- J-DSP Problem 2.1: The student is required to design and simulate in J-DSP a discrete-time system with an exponential impulse response. LEARNING: The student learns how to use J-DSP to invert z-transforms and obtain computationally the time-domain signal. The J-DSP visualization exercise reinforces the relation of z-plane singularities with filter stability.
- J-DSP Problem 2.2: The student is required to design and simulate using J-DSP a digital oscillator. LEARNING: The student learns how digital oscillators are designed and implemented. Digital (software) oscillators are used in cell phones to generate the dialing (DTMF) tones.
- J-DSP Problem 2.3: The student is required to design and simulate using J-DSP an FIR filter that cancels sinusoidal interference. LEARNING: The student learns how an FIR filter can be configured to cancel 60Hz interference.
- J-DSP Problem 2.4: The student is required to simulate a linear-phase filter. LEARNING: The student learns how an FIR linear phase system behaves in the time-domain, z-domain, and frequency domain. Such filters are used in Digital Hi-Fi audio systems.
- J-DSP Problem 2.5: The student is required to design and simulate a filter with a prescribed pole-zero locations. LEARNING: The student learns inter-relationships of z-plane and Fourier transforms.
- J-DSP Problem 2.6: The student is required to simulate cascade and parallel configurations of digital filters. LEARNING: The student learns the advantages of cascade and parallel configurations and associates these schemes with the theory of linear systems.

A complete copy of the above laboratory exercise is in Appendix D-1. Assessment of this J-DSP exercise is presented in Appendix C–1 and in reference [6].

4.2 J-DSP Software User Manual

J-DSP is a web-based simulation environment that can be used for several purposes. To assist students and other casual users to use J-DSP, we have prepared several documents (included in Appendix F), i.e.,

- a complete manual where each function or block is described (Appendix F),
- An introductory section that explains how to use *drag n' drop* process to establish simulations,
- A section that shows how to use *J-DSP scripts* to integrate J-DSP simulations in HTML content,
- A *Quick Reference / Getting Started Guide* included in the dissemination package,
- Several AVI files that have been posted on the web site to show how the software works.

5. J-DSP Labware Assessment

J-DSP assessment was carried through the years as part of the evaluation of the EEE 407 class. Specific evaluation for J-DSP was also carried including: evaluations by web site visitors, required student evaluations of each lab after EEE407 students submit their reports, and specific *differential* evaluations of each J-DSP lab with pre- and post- lab tests that measure the degree of learning specifically attributed to the usage of J-DSP. We include the following documents in the NEEDS package

- An assessment report is included in Appendix C-1.
- Sample assessment forms are shown at the end of Appendix C-1.
- Specific comments by EEE 407 students are on Appendix C-2.
- Lists of faculty that received J-DSP and agreed to form beta sites are included in Appendix C-3. A copy of the dissemination package that was distributed freely at the 2002 IEEE DSP Workshop in Atlanta and IEEE/ASEE FIE-02 in Boston is also included. Reference letters of faculty and industry engineers that agreed to evaluate the software are in Appendix B.

Although initially J-DSP assessment was carried using the course evaluation forms, during the last academic year we assessed separately the degree of learning attributed specifically to J-DSP. We focused in particular on assessing whether J-DSP accelerated the learning process in the DSP class. Both on-line and off-line instruments have been developed. Two types of on-line forms have been developed, i.e., general and concept-specific. In the general forms, the students are asked to provide general qualitative and quantitative evaluation of J-DSP and information collected includes comments on logistics, accessibility, convenience, demographics, academic standing, etc. In concept-specific forms, students evaluated each laboratory task with regard to its impact on learning specific DSP concepts. Our newest assessment instruments assess learning attributed specifically to J-DSP by testing DSP concepts before (pre-assessment) and after (post-assessment) the J-DSP lab. Assessment results have recently been submitted to the Online Evaluation Resource Library (OERL).

From the evaluation document in Appendix C-1, we see that most of the students (almost 90% or above) agreed that the J-DSP labs helped them understand the DSP related concepts. In pre- and post-assessment (Fig. 2) students were given a quiz in class prior to performing the J-DSP laboratory and the same quiz was administered after a lab was performed. We observe that students have performed better in the assessment quiz after they have completed their J-DSP labs. Questions 1 and 2 assessed whether students can relate the pole and zero positions to the resulting magnitude frequency response. A 13.1% improvement was realized in Question 1 and a 13.29% was realized in question 2 in the post lab assessment. In Question 3, two polynomial functions are given and students are asked if their magnitude responses are equal. A 10.99% improvement is noticed here. In Question 4, students are asked about the filter type (i.e. low-pass/high-pass etc.) of a given transfer function. We see that only 28.26% of students answered correctly in pre-lab assessment and it is indeed poor but the percentage has increased by 44.71% in the post-lab assessment. In Question 5, students are given four choices regarding the change of magnitude response with displacement of the zeros with respect to the unit circle. A noticeable improvement of 25.62% in the post-lab quiz is evident.

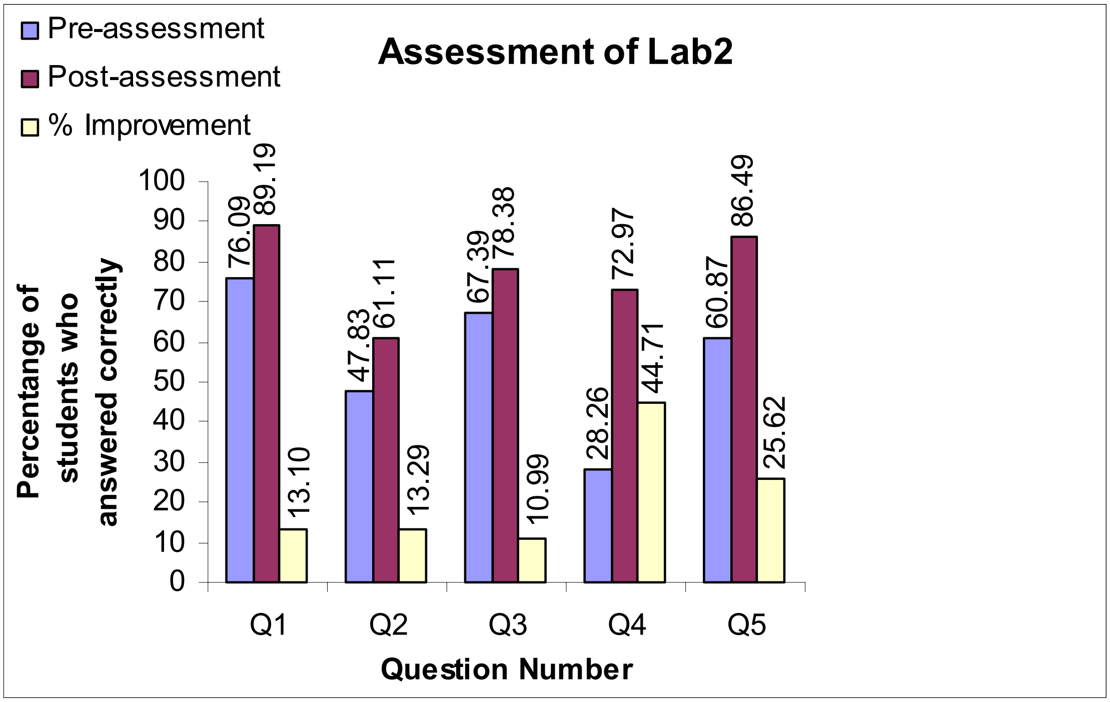


Fig.2. Sample Results on Pre- and Post Assessment of J-DSP Laboratory 2
(for more results on all J-DSP labs see Appendix C-1)

5.1 General Remarks on Assessment

Continuous feedback helped correct and improve the J-DSP software and exercises. The students in general found the J-DSP concept very convenient and easy to use. Concept specific assessments and pre/post assessment revealed that several J-DSP functions have been particularly useful in communicating key DSP concepts. J-DSP was proven to be particularly useful in learning issues related with filter design and interpretation of frequency spectra. The J-DSP visualizations involving pole-zero diagrams have shown prominent differences in pre- and post- assessments that lead us to believe that we need to integrate even more animation and develop demonstrations that are dynamic. In all, EEE 407 students asserted that they have benefited from J-DSP and they particularly appreciated the fact that the tool was available on the web from any location. Industry students taking the course from remote sites have been particularly impressed with the tools and exercises and some are using it for routine design and other compact DSP simulations.

6. Endorsements

6.1 Endorsements from Colleagues

Several colleagues have obtained and endorsed J-DSP software and recommendation letters are attached at the end of this document. Comments from recommendation letters are itemized below:

- Professor John Proakis, who is one of the most successful text book authors in communications and signal processing, writes "*I have been impressed by several features of J-DSP and the fact that this software, that is essentially an entire programming system with an interpreter, was developed by faculty and students with minimal resources.*"
- Professor Paul Hasler from Georgia Tech writes "*J-DSP is indeed a unique software tool that will impact many of the courses related to linear systems and signal processing.*"

- Professor C. L. Max Nikias, Dean of the School of Engineering at USC writes "*This software essentially enables DSP courseware with embedded on-line simulations and establishes a new paradigm for running DSP simulations and laboratories over the Internet.*"
- Prof. Antonia Papandreou-Suppappola of Arizona State University writes "*The J-DSP software provides a unique active learning methodology for digital signal processing. As a result, we started to use some of its functionalities in other Systems area courses such as Communications Systems and Advanced Signal Processing courses."*
- Professor Tolga Duman of Arizona State University writes "*The students and the instructors in this course believe that this on-line tool is extremely important for student learning, specifically, student evaluations of the software have been consistently positive.*"
- Professor G. Faye Boudreaux-Bartels of the University of Rhode Island writes *"I have experimented with your J-DSP laboratories for senior undergraduate and graduate students and found them very useful."*
- Professor Brian Evans from the University of Texas-Austin writes *"your original object oriented data flow Java-DSP environment will be extremely useful in enabling the students to explore and associate various important engineering applications with the abstract mathematical concepts taught in these junior level courses."*

Complete recommendation letters and agreements to host and use J-DSP are given in the Appendix B.

6.2 Endorsements from Professional Societies

The IEEE Phoenix section awarded Professor Spanias with the Educator Award of the Year 2003 "for his contributions in the J-DSP project."

6.3 Endorsements from Students

- Y. Song, a former EEE 407 student writes "the graphical user interface helps the user understand some basic DSP concepts better."
- C. Panayiotou, a former EEE 407 student writes "the topic regarding the effect of windows on FFT spectra became clear with exercises and simulations conducted using J-DSP."
- A. Natarajan a former EEE 407 student "I particularly liked that one could move the poles and zeros around the unit circle and observe the corresponding effect on the peaks and valleys present in the frequency response."
- Several anonymous student comments collected from J-DSP web evaluations are itemized in Appendix C-2.

7. J-DSP *Labware* and NEEDS Courseware Criteria

J-DSP is not multimedia courseware per se and therefore some items from the NEEDS criteria may not apply in the J-DSP NEEDS submission. We classify J-DSP as *labware*, i.e., software that enables students to perform educational laboratories on the web. We believe that this software must be evaluated by NEEDS because it represents a technology innovation that is intended solely for education and addresses the laboratory aspect of education that perhaps not many other submissions address. Because this submission is not the traditional multimedia courseware, we submitted all the materials supporting J-DSP to demonstrate to the reviewers that J-DSP is a comprehensive effort with a large volume of supporting text, manuals, assessment, and dissemination materials. We will be grateful if the reviewers in their evaluation take the time to look at our submitted Appendices that demonstrate well our commitment to this educational task. To our knowledge, we are the first to submit assessment of labware to the OERL database sponsored by NSF.

7.1 Instructional Design

7.1.1 Learning Objectives:

Our learning objectives have been stated in Section 3 of this submission packet. The assessment of our objectives has been summarized in Section 5 and is detailed further in Appendix C. The students perform a sequence of laboratory exercises (Appendix D) using the J-DSP software. The exercises have been designed carefully to support the learning of the topics covered in the DSP class that range from discrete-time filter basics to filter design and spectral analysis. By providing hands-on experiences on the class topics J-DSP enables the student to associate abstract mathematical concepts of DSP theory to actual implementation issues.

7.1.2 Support and Measurement of Learning Objectives.

Because the laboratory assignments are well coordinated with in-class EEE 407 lectures that are available to students either real-time or through synchronous and asynchronous webcasting, the students are well aware of the learning objectives for each topic. From assessments (Appendix C) it is evident that J-DSP enabled the students to comprehend and digest several topics that were not immediately obvious by attending the class or reading the text book.

7.1.3 J-DSP Learning objectives and ABET accreditation criteria.

We have carried a self study on EEE 407 and re-evaluated how the course and particularly its J-DSP laboratory addresses ABET criteria. With regard to ABET design requirements, students are asked to solve open-ended problems as part of the J-DSP laboratory exercises. The J-DSP lab involves extensive filter design using several methods and students are asked to analyze these filter design methods and perform comparisons. Students are asked to design algorithmic steps and develop J-DSP realizations of these algorithmic steps. *More specifically the J-DSP laboratory component addresses well ABET Criteria 3 and 8.* The J-DSP laboratory tasks in almost all the J-DSP labs described in Appendix D engage the students into applying their knowledge of mathematics and engineering (*ABET 3a*). For example in J-DSP Labs 2 and 5 students apply the mathematical properties of Fourier transforms and z transforms to real world engineering applications such as filtering and spectral analysis (relates to ABET 3a). There are specific labs that involve filter design (J-DSP Lab 4 in Appendix D) where the student is required to use J-DSP to perform several designs and compare and choose the one that satisfies specific criteria (*ABET 3b*). *ABET 3c* relates to J-DSP lab design tasks such as the design of linear phase filters (J-DSP Lab 4), digital oscillators (J-DSP Lab 2), and QMF filter banks (J-DSP Lab 6). Several engineering problems are solved within the context of J-DSP labs relating to *ABET 3e*. J-DSP tasks also satisfy *ABET Criterion 8* since discrete mathematics and linear algebra are inherent to DSP.

7.2 Interactivity

The user-friendly interface of J-DSP allows simulations to be performed interactively. The user can examine every signal at any part of the algorithm block diagram. By double clicking on a block, a dialog window appears that allows interaction and modification of various parameters. In EEE 407 J-DSP enables the user to go through a variety of laboratory exercises that cover several aspects of DSP theory. With regard to two way communication, it is possible for the instructor to develop web and streaming content where feedback on J-DSP tasks is given automatically. Although we have in place the J-DSP capabilities for scripting that allow integration of J-DSP demos with web content and multimedia presentations, we have not yet finalized a formal web class. We are currently in the midst of producing a DSP web class with streaming content that will incorporate interactive sessions with J-DSP. The goal is to eventually produce an environment similar to that shown in Appendix A – Fig. A.5.

7.3 Application of Concepts to other areas

When students complete the J-DSP lab sequence they are required to propose and complete a project that relates to a real world application. Such projects include models for compression algorithms used in cellular phones, MP3, and JPEG. Our class evaluations reveal that J-DSP is responsible for building the necessary programming and algorithmic intuition required to perform DSP algorithm projects. By the very nature of the J-DSP environment and the assigned open ended lab exercises, students develop the ability to explore multiple solutions to design problems and extend ideas to other application areas. We also have evidence that many of the 407 students attend graduate school and engage in algorithm development in their M.S. theses.

7.3.1 Cognition/Conceptual Changes

Our DSP class and particularly our pre- and post-assessment (Appendix C) reveal that J-DSP enhances learning of the key topics covered in the DSP class. Statistics and details for each lab are given in Appendix C. In the J-DSP laboratory students submit electronic reports. Portions of the report are graded automatically and students get instant feedback.

7.4 The content structure

The content in the J-DSP lab exercises (Appendix D) is structured in a manner that follows the progression of topics in any undergraduate DSP course. The next laboratory exercise builds on knowledge gained by the previous one. The software functions (blocks) in J-DSP are also organized in an intuitive manner. Frequently used functions always appear to the left (filter, Sig.Gen, etc). The rest of the functions are grouped in terms of filter, frequency domain, statistical etc (see manual in Appendix F). The number of functions available in J-DSP exceeds those required in the labs and several scripted J-DSP demos are available for the student to explore additional topics not only in DSP but also in other areas. Explanations on these J-DSP demos are available on the web site. Students are also encouraged to verify theory using J-DSP and several J-DSP simulations are run in class to demonstrate advanced topics.

7.5 Multimedia and J-DSP

J-DSP offers a variety of visual and audio tools to enhance learning. Student feedback has shown that users of J-DSP achieved learning using J-DSP animations and the J-DSP sound player. The sound player is used in many cases to reinforce concepts in filtering and signal compression. For example, we provide functions that connect music to signals and spectra and we promote these J-DSP functions for early exposition to DSP in high schools (see paper [7]). The very nature of Java, which is the backbone of J-DSP, allows for integrated animations in web and streaming content. Functions to integrate more and more multimedia are continuously being developed.

7.6 Instructional Use

Engineering students can consult a detailed J-DSP manual that is posted on the internet. Every function has explanations and an integrated help screen. The J-DSP support web site has integrated assessment tools, a way to report problems, frequently asked questions site, and plenty of support. Forms are provided for assessment and feedback and student/user responses and associated statistics are automatically posted (using MS *asp* technologies) on the web site. These can be viewed on the J-DSP web site (http://jdsp.asu.edu).

This software enables instructors to integrate simulations and Java animations in their web content. The tedious Java programming of DSP tasks is replaced by object oriented visual programming. That is, the instructor can generate easily J-DSP scripts that have been developed to activate J-DSP simulations from HTML (see Appendix A - Fig. A3). *We view this as one of the greatest potentials for this tool.* Expansion of J-DSP used to other areas such as communications, controls, and image processing has already started [11]. We are also embedding this software in a new web course on DSP that we are developing.

8. Software Design

The software is reasonably engaging when used along with the lab exercises and does not contain stereotypes. Although, it would take sometime to load when connected with a POTS (33k) modem, once J-DSP is loaded the execution is quite fast. Loading is also very fast with ISDN, DSL, LAN, and cable modems. The software is continuously improved and we feel strongly that the DSP portion of J-DSP is reasonably free of bugs. One of the challenges in maintaining J-DSP is dealing with the rapidly changing technologies of web browsers. Therefore constant maintenance is essential.

8.1 Learner Interface and Navigation

There are only a few basic repetitive steps one needs to perform in order to construct simulations with J-DSP. By pressing a button the user can select and then drag-and-drop any block where desired. Connections can be constructed simply by dragging the mouse from a block's output to another block's input. Last but not least, all blocks have a dialog window that opens when the user double clicks on it. All dialog windows are consistent and include a Help facility.

9. Engineering Content

9.1. Accuracy

The software has proven to be accurate and quite stable. Before releasing it for use, a series of tests have been performed to validate the correctness of the code. Procedures are in place to report and correct bugs.

9.2 Appropriateness

J-DSP is an educational on-line simulation environment developed to support labs. Although it is not the traditional multimedia courseware type, it falls within the context of software developed for education by educators and we hope that the NEEDS panel will evaluate this novel *labware* concept for the NEEDS award.

Bibliography (copies of the publications have been included in the same order in Appendix E)

[1] Clausen **A.,** Spanias A., Xavier A. "A Java Signal Analysis Tool for Signal Processing Experiments, *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP-98)*, DSP 16, Seattle, May 1998.

[2] Spanias A. et al, "Development of a Web-based Signal and Speech Processing Laboratory for Distance Learning," *ASEE Computers in Education Journal*, pp. 21-26, Vol. X, No.2, April-June 2000.

[3] Spanias A. and Bizuneh F., "Development of new functions and scripting capabilities in java-dsp for easy creation and seamless integration of animated dsp simulations in web courses**,"** *Proc. IEEE International Conference on Acous.c, Speech and Sign. Proc. (ICASSP-2001),* pp. 2717-20, Salt Lake City, May 2001.

[4] Thrasyvoulou T., Tsakalis K. and A. Spanias, "J-DSP-C, A Control Systems Simulation Environment for Distance Learning: Labs and Assessment," 33rd ASEE/IEEE FIE-03 Conf, Boulder, Nov 5-8, 2003.

[5] Zaman M., Papandreou-Suppappola A. and Spanias A "Advanced Concepts in Time-Frequency Signal Processing made Simple," 33rd ASEE/IEEE FIE-03, Boulder, Nov 2003.

[6] Spanias A., Ahmed K., Papandreou-Suppappola A., and Zaman M., "Assessment of the Java-DSP (J-DSP) On-Line Laboratory Software," 33rd ASEE/IEEE FIE-03, Boulder, Nov 2003.

[7] Spanias A., T. Thrassyvoulou, C. Panayiotou, Y. Song, "Using J-DSP to Introduce Communications and Multimedia Technologies to High Schools," 33rd ASEE/IEEE FIE-03, Boulder, November 2003.

[8] Yasin M., Karam L., and Spanias A., "On-Line Laboratories For Image And Two-Dimensional Signal Processing," 33rd ASEE/IEEE FIE-03, Boulder, Nov 2003.

[9] Atti V. and Spanias A., "On-line Simulation Modules for Teaching Speech and Audio Compression," 33rd ASEE/IEEE FIE-03, Boulder, Nov 2003.

[10] Ko, Y. Duman, T., Spanias A., "J-DSP for Communications," 33rd ASEE/IEEE FIE-03, Boulder, Nov. 2003

[11] Spanias A. et al, "On-Line Laboratories for Speech and Image Processing and for Communication Systems Using J-DSP," IEEE 2002 DSP Workshop, Callaway, Georgia, October 2002.

Northeastern
UNIVERSITY

College of Engineering
Department of Electrical and Computer Engineering
440 Dana Research Center
Northeastern University
Boston, Massachusetts 02115-5000

Phone: 617.373.4159
Facsimile: 617.373.8970

To: The NEEDS award committee,
From: Professor John Proakis
Date: 6/4/03
Subject: Recommendation letter of the J-DSP educational software

This is regarding the educational Java software J-DSP that was developed by Prof. Andreas Spanias and his team at Arizona State University. I became aware of J-DSP at the 2002 IEEE/ASEE Frontiers in Education conference in Boston where J-DSP was freely disseminated to Engineering educators. I received a free copy of this software for evaluation.

The J-DSP software was developed solely for instruction at Arizona State University with funds from the state of Arizona and the National Science Foundation. The objective of the original J-DSP project was to develop a programming environment to provide computer laboratory experiences over the internet to distance learners. The outcome of this project was an object-oriented DSP simulation environment. The J-DSP simulation environment can run from any location and from any platform equipped with an internet browser. The modem speed and computer requirements are modest and therefore it proved to be quite useful and cost effective for distance learning.

The original version of J-DSP was developed in 1995 and results have been published at the 1997 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP-97). Originally the software accommodated only basic signaling, filtering, and FFT-based simulations. The software received immediate attention from DSP educators and was posted on several web sites worldwide. Since that time, the software has been continuously augmented with new DSP functionality. Several changes also have been implemented on the infrastructure of the program.

The purpose of the J-DSP software was non-commercial and there has been no attempt to commercialize it. In fact the software specializes on compact educational simulations and does not compete with products of The Mathworks or packages such as MathCad. ASU started using this software in their undergraduate course and Andreas Spanias has developed and published on the web a suite of computer exercises whose aim was to provide hands-on experiences that complement the theory taught in his DSP classes. The program became very popular particularly with distance learners.

Andreas Spanias and his team developed several functions that not only render this software as a simulation tool but also as an instruction tool. The functionality developed includes

- J-DSP scripting that enables instructors to built Java simulations for DSP and embed them in the web content

- An interface to MATLAB that enables instructors to port results from MATLAB simulations and embed them in J-DSP simulations and on the web

- Demonstration functionality that enables instructors to built demos of DSP algorithms and include them in web pages without engaging into tedious Java programming.

The NSF project funded in the year 2000 enabled the ASU team to expand the functionality of this software to cover other systems courses such as

- Analog and Digital Communications systems
- Statistical signal processing
- Speech processing
- Image processing and 2-D DSP
- Controls systems

On a personal note, I have been impressed by several features of J-DSP and the fact that this software, that is essentially an entire programming system with an interpreter, was developed by faculty and students with minimal resources. I find the J-DSP graphical interface to be user friendly and intuitive. The concept of building a simulation and establishing signal flow using block diagrams by using a browser is easy to learn and the programming concept is consistent with the way we view signal processing and *communications systems* as educators and engineers. According to J-DSP assessment results, which will appear in the IEEE/ASEE FIE-03 proceedings, students reported that it took most of them less than fifteen minutes to become familiar with running simulations on J-DSP. This was without use of a manual and by just using a small introduction in their first DSP exercise on the web. In the same paper, pre- and post-assessment results revealed that the J-DSP tool helped students understand several concepts that were not immediately evident from their text book or class lectures. In fact the study showed that in some cases learning was attributed exclusively to using J-DSP.

There are a total of seven papers that will be presented in this year's FIE-03 highlighting different aspects of the J-DSP software, including extensions in communications, controls and image processing and recently functions that can be used to introduce high-school students to the processing and interpretations of everyday signals. The papers are co-authored by Andreas Spanias and his collaborators and are listed below:

- "A Control Systems Simulation Environment For Distance Learning: Labs And Assessment," 33rd ASEE/IEEE FIE Conference, Boulder, CO, Nov 5-8,2003.
- "Advanced Concepts in Time-Frequency Signal Processing made Simple," 33rd ASEE/IEEE FIE-03, Boulder, November 2003
- "Assessment of the Java-Dsp (J-DSP) On-Line Laboratory Software ," 33rd ASEE/IEEE FIE-03, Boulder, November 2003
- "Using J-DSP to Introduce Communications and Multimedia Technologies to High Schools," 33rd ASEE/IEEE FIE-03, Boulder, November 2003
- " On-Line Laboratories For Image And Two-Dimensional Signal Processing ," 33rd ASEE/IEEE FIE-03, Boulder, November 2003
- " On-line Simulation Modules for Teaching Speech and Audio Compression ," 33rd ASEE/IEEE FIE-03, Boulder, November 2003

- "J-DSP for Communications," 33rd ASEE/IEEE FIE-03, Boulder, November 2003

The J-DSP program now features advanced functions such as linear prediction, Hidden Markov models, vector quantization, channel equalization, multi-rate QMF banks, and so on. Modules and streaming lectures are being prepared to introduce even undergraduate students to these topics.

I applaud the efforts of Andreas Spanias and his team at Arizona State. J-DSP is an exemplary concept and truly worthy of the NEEDS award.

Sincerely,

John Proakis

John G. Proakis, IEEE Fellow
Former Chair and Professor Emeritus, Northeastern University

USC
UNIVERSITY
OF SOUTHERN
CALIFORNIA

School of Engineering

Office of the Dean

C.L. Max Nikias
Zohrab A. Kaprielian
Dean's Chair in
Engineering

June 9, 2003

The NEEDS Award Committee:

This is written in support of the J-DSP software that is being considered for the NEEDS award. The Java-DSP (J-DSP) software has been developed from the ground up for education at Arizona State University and is perhaps the first on-line signal processing simulation tool. The purpose of this educational tool , which has been originally presented at ICASSP-97 in Munich-Germany, is to enable distance learners to perform on-line laboratories on any platform and from any location. J-DSP in fact established a new paradigm for distance education where computer laboratories and DSP algorithm simulations are performed on the web using a simple web browser. J-DSP simulations are established using a Java-based object oriented environment where blocks represent DSP algorithms and when blocks are connected signal flow is established and simulations with multiple algorithms execute.

This educational tool is being used to facilitate the laboratory portion of the DSP course at Arizona State for the last six years. This course has a distance learning audience and the presence of a web-based laboratory became very popular among the students particularly those at remote locations. A series of laboratories have been developed by Prof. Spanias as a companion to this exemplary software tool. Electronic web-based submission software consisting of sevlets was also developed at ASU to make the lab-report submission and grading web-based. To this date the students at ASU use J-DSP and the web-based submission system in their DSP class. The original J-DSP concept and the submission system have been described by Prof. Spanias and his team in the ASEE Computers in Education Journal, pp. 21-26, Vol. X, No.2, April-June 2000.

The DSP prototype was originally funded by Arizona State funds and subsequently extensions to this prototype have been funded by NSF. The functionality of this software has been extended to cover other topics such as communications and controls. In addition, the infrastructure of the software environment was enhanced to provide additional features that made it very useful to DSP instructors. The software was formally disseminated at the 2002 IEEE DSP workshop and at the

University of
Southern California
Los Angeles.
California 90089-1450
Tel: 213 740 0617
Fax: 213 740 8493
e-mail:
engrdean@usc.edu
web page:
http://www.usc.edu/
dept/engineering

FIE conference in Boston in the fall of 2002. Approximately 50 university professors obtained evaluation copies from universities including MIT, Georgia Tech, Rice University, U. Minnesota, and UPenn. In all there are about fifty worldwide beta sites that are being established, including some in major universities in Argentina, China, Finland, Puerto Rico, Portugal, Spain, Sweden, Switzerland, Turkey, and the United Kingdom.

Functionality extensions have been reported by Prof. Spanias and his team at ICASSP 2001. A major capability was developed for this tool by designing a script interpreter that enables users to embed interactive J-DSP simulations in html content. This particular capability enables instructors to design web courses with interactive visualization content that is developed using J-DSP. The value of this feature is that instructors do not need to engage in tedious Java programming to embed simulations in their web content. Other functionality was also developed to enable advance algorithms to execute in J-DSP. Furthermore assessments of learning attributed specifically to the use of this software was carried on the web. Functionality extensions, formal assessment, and implementation of advance algorithms are being reported this year in a series of publications at IEEE/ASEE FIE 03.

I highly recommend J-DSP educational software for the NEEDS award. This software essentially enables DSP courseware with embedded on-line simulations and establishes a new paradigm for running DSP simulations and laboratories over the internet.

Sincerely,

C.L.Nikias

C. L. Max Nikias

Georgia Tech College of Engineering
School of Electrical and Computer Engineering

June 10, 2003

To whom it may concern:

This letter is written in support of the Java-DSP (J-DSP) educational software that has been nominated for the 2003 NEEDS award. The J-DSP on-line simulation software was freely disseminated at the 2002 IEEE DSP workshop. Georgia Tech obtained a free copy for testing and colleagues and students are currently evaluating this software. The idea with J-DSP is to provide a free programming and visualization environment for students and instructors. This java-based software provides a visual simulation environment that is accessible on the internet through a simple browser; the original idea with J-DSP has been to enable on-line computer laboratories. Professor Andreas Spanias has done an excellent job incorporating J-DSP into his undergraduate curriculum, since J-DSP is being used routinely now in the undergraduate DSP course at Arizona State University with an on-line laboratory manual with exercises. Instructors can built Java-based DSP simulations and embed them in their courseware.

Articles describing J-DSP have been published in conferences and journals. Several papers have been presented in engineering education conferences and special sessions. These papers describe new functionality in developed for J-DSP as well as assessment results obtained from using the software in on-line laboratories delivered in distance learning classes. In the DSP workshop a paper on functionality extensions that support communications, controls, speech processing, and image processing has been presented.

J-DSP is indeed a unique educational software tool that will impact many of the courses related to linear systems and signal processing. We enthusiastically support the nomination of this unique on-line laboratory software for the NEEDS award.

Sincerely

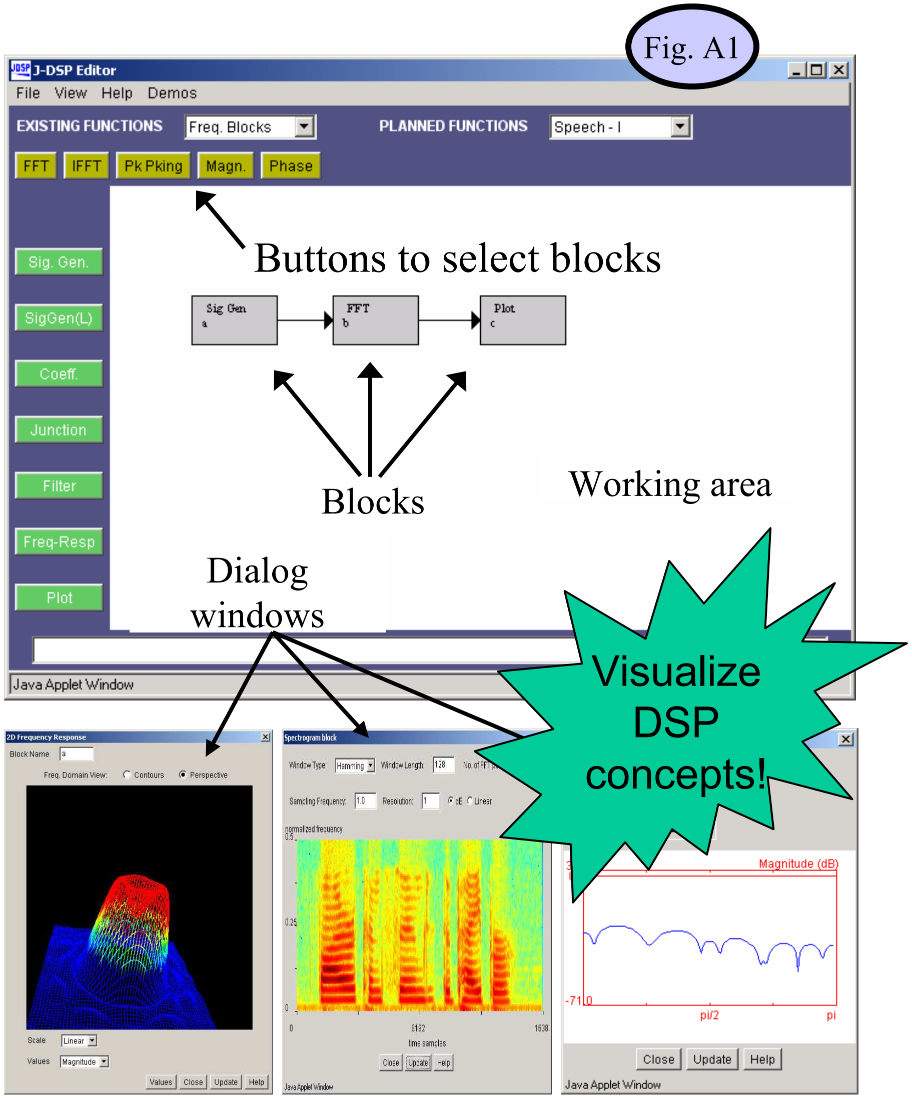
Dr. Paul Hasler,
Associate Professor
Department of Electrical and Computer
Engineering
Georgia Institute of Technology

J-DSP project at a glance

J-DSP®
Editor

J-DSP: A new paradigm in education!

- J-DSP is an on-line, object-oriented graphical DSP editor written as a Java applet
- It creates a new paradigm for education
- It can be used to simulate DSP systems
- Users can view the results at any point of the simulation, graphically or numerically
- It provides a simple and user-friendly interface



EXISTING FUNCTIONS: Freq. Blocks

PLANNED FUNCTIONS: Speech - I

Block Name: a

Freq. Domain View: ○ Contours ● Perspective

Scale: Linear

Values: Magnitude

Window Type: Hamming Window Length: 128 No. of FFT p

Sampling Frequency: 1.0 Resolution: 1 ● dB ○ Linear

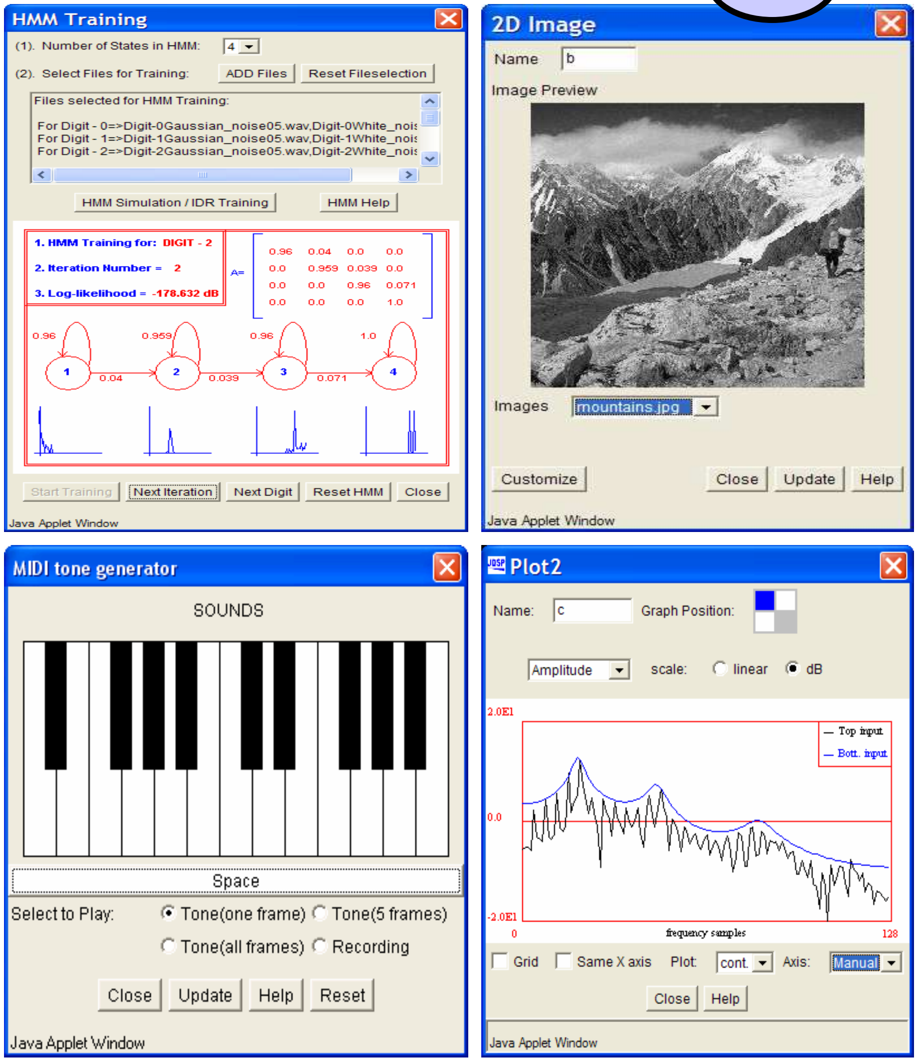
J-DSP

Existing functionality

J-DSP Supports:

- IIR & FIR Filter Design
- Multirate DSP
- Adaptive Filtering
- Spectral Estimation
- Simulating Vocoders (CELP, LPC)
- Fundamental DSP Functions (FFT, IFFT, Windowing etc.)

Fig. A2



HMM Training

(1). Number of States in HMM: 4

(2). Select Files for Training:

Files selected for HMM Training:

For Digit - 0=>Digit-0Gaussian_noise05.wav,Digit-0White_nois
For Digit - 1=>Digit-1Gaussian_noise05.wav,Digit-1White_nois
For Digit - 2=>Digit-2Gaussian_noise05.wav,Digit-2White_nois

1. HMM Training for: DIGIT - 2

2. Iteration Number = 2

3. Log-likelihood = -178.632 dB

$$A=\begin{bmatrix} 0.96 & 0.04 & 0.0 & 0.0 \\ 0.0 & 0.959 & 0.039 & 0.0 \\ 0.0 & 0.0 & 0.96 & 0.071 \\ 0.0 & 0.0 & 0.0 & 1.0 \end{bmatrix}$$

2D Image

Name: b

Images: mountains.jpg

MIDI tone generator

Select to Play: ● Tone(one frame) ○ Tone(5 frames) ○ Tone(all frames) ○ Recording

Plot2

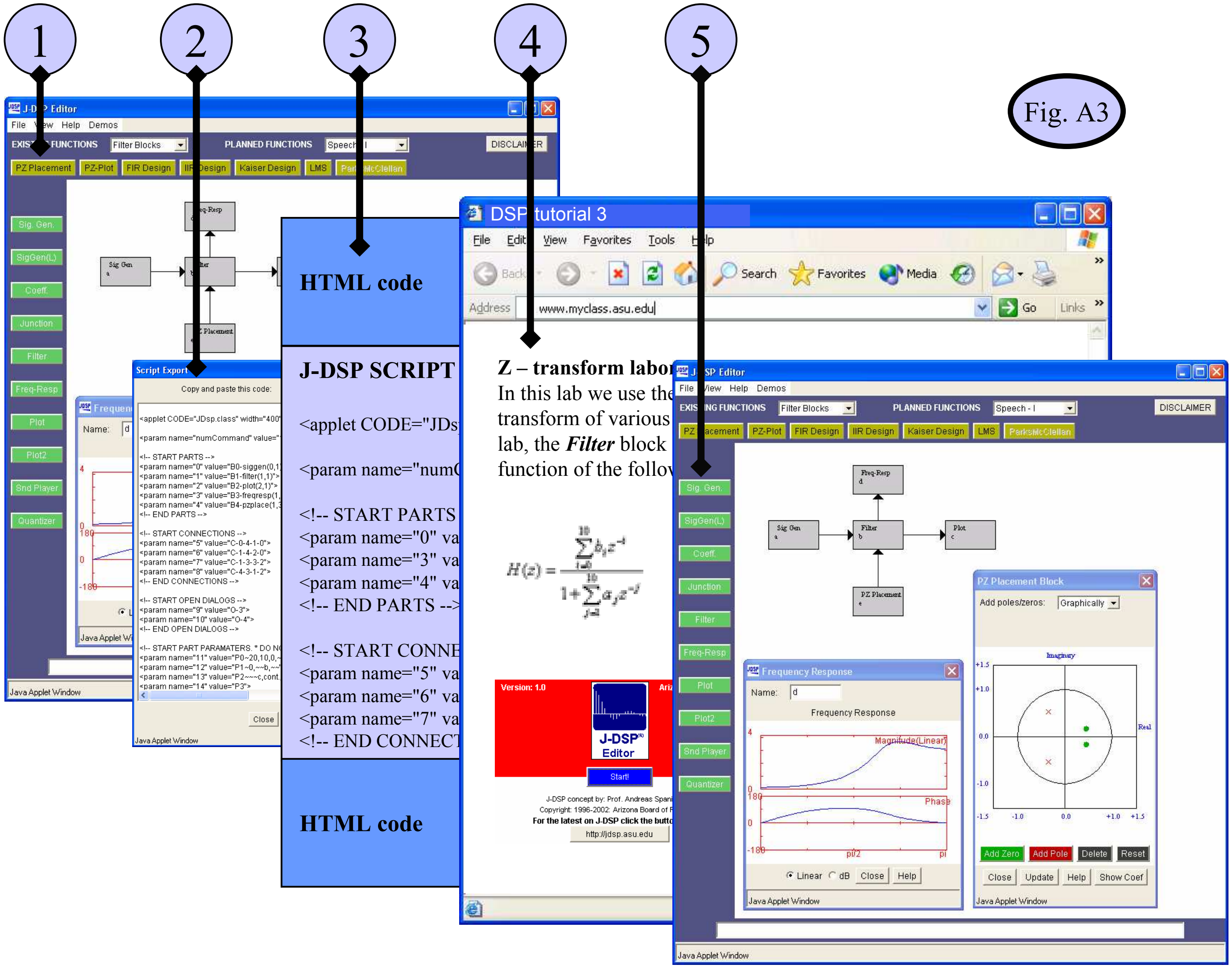
Name: c

Amplitude scale: ○ linear ● dB

□ Grid □ Same X axis Plot: cont. Axis: Manual

Seamlessly embed J-DSP simulations in web content

Fig. A3



```
<applet CODE="JDsp.class" width="400"

<param name="numCommand" value="

<!-- START PARTS -->
<param name="0" value="B0-siggen(0,1)
<param name="1" value="B1-filter(1,1)">
<param name="2" value="B2-plot(2,1)">
<param name="3" value="B3-freqresp(1,
<param name="4" value="B4-pzplace(1,3
<!-- END PARTS -->

<!-- START CONNECTIONS -->
<param name="5" value="C-0-4-1-0">
<param name="6" value="C-1-4-2-0">
<param name="7" value="C-1-3-3-2">
<param name="8" value="C-4-3-1-2">
<!-- END CONNECTIONS -->

<!-- START OPEN DIALOGS -->
<param name="9" value="O-3">
<param name="10" value="O-4">
<!-- END OPEN DIALOGS -->

<!-- START PART PARAMATERS. * DO NO
<param name="11" value="P0~20,10,0,~
<param name="12" value="P1~0,~~b,~~
<param name="13" value="P2~~~c,cont.
<param name="14" value="P3">
```

www.myclass.asu.edu

Z – transform labor
In this lab we use the
transform of various
lab, the ***Filter*** block
function of the follov

$$H(z)=\frac{\sum_{i=0}^{10} b_i z^{-i}}{1+\sum_{j=1}^{10} a_j z^{-j}}$$

J-DSP concept by: Prof. Andreas Spani
Copyright: 1996-2002: Arizona Board of R
For the latest on J-DSP click the butto
http://jdsp.asu.edu

1. Prepare demonstration in J-DSP. **2**. Export simulation in J-DSP script. **3**. Copy and paste script into an HTML file.
4. Add your own educational content **5**. Deliver to students.

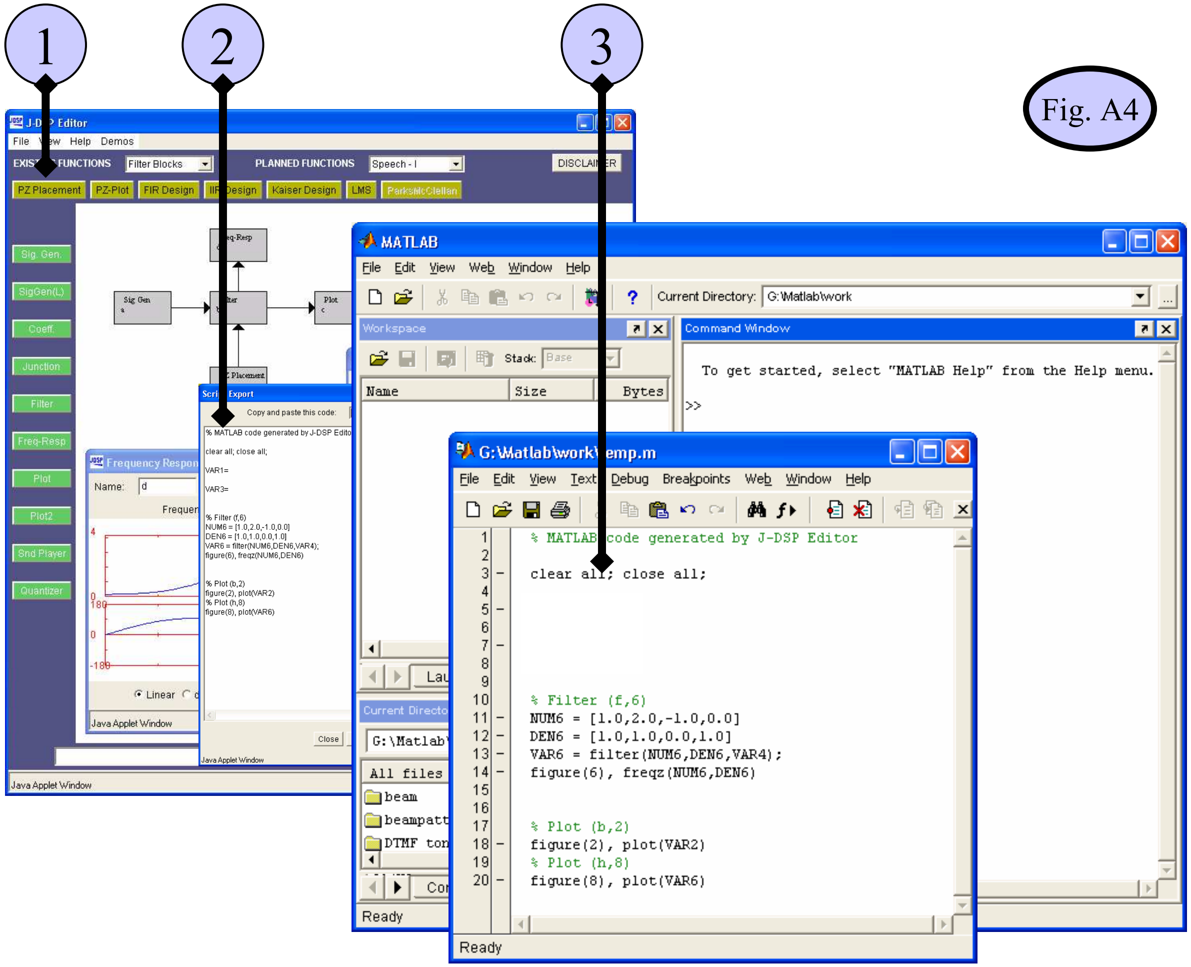
J-DSP

Future functionality

(Proposed to NSF)

Export simulations in MATLAB™ code

Fig. A4



Script Export

Copy and paste this code:

```
% MATLAB code generated by J-DSP Edito

clear all; close all;

VAR1=

VAR3=


% Filter (f,6)
NUM6 = [1.0,2.0,-1.0,0.0]
DEN6 = [1.0,1.0,0.0,1.0]
VAR6 = filter(NUM6,DEN6,VAR4);
figure(6), freqz(NUM6,DEN6)


% Plot (b,2)
figure(2), plot(VAR2)
% Plot (h,8)
figure(8), plot(VAR6)
```

Close

```
To get started, select "MATLAB Help" from the Help menu.

>>
```

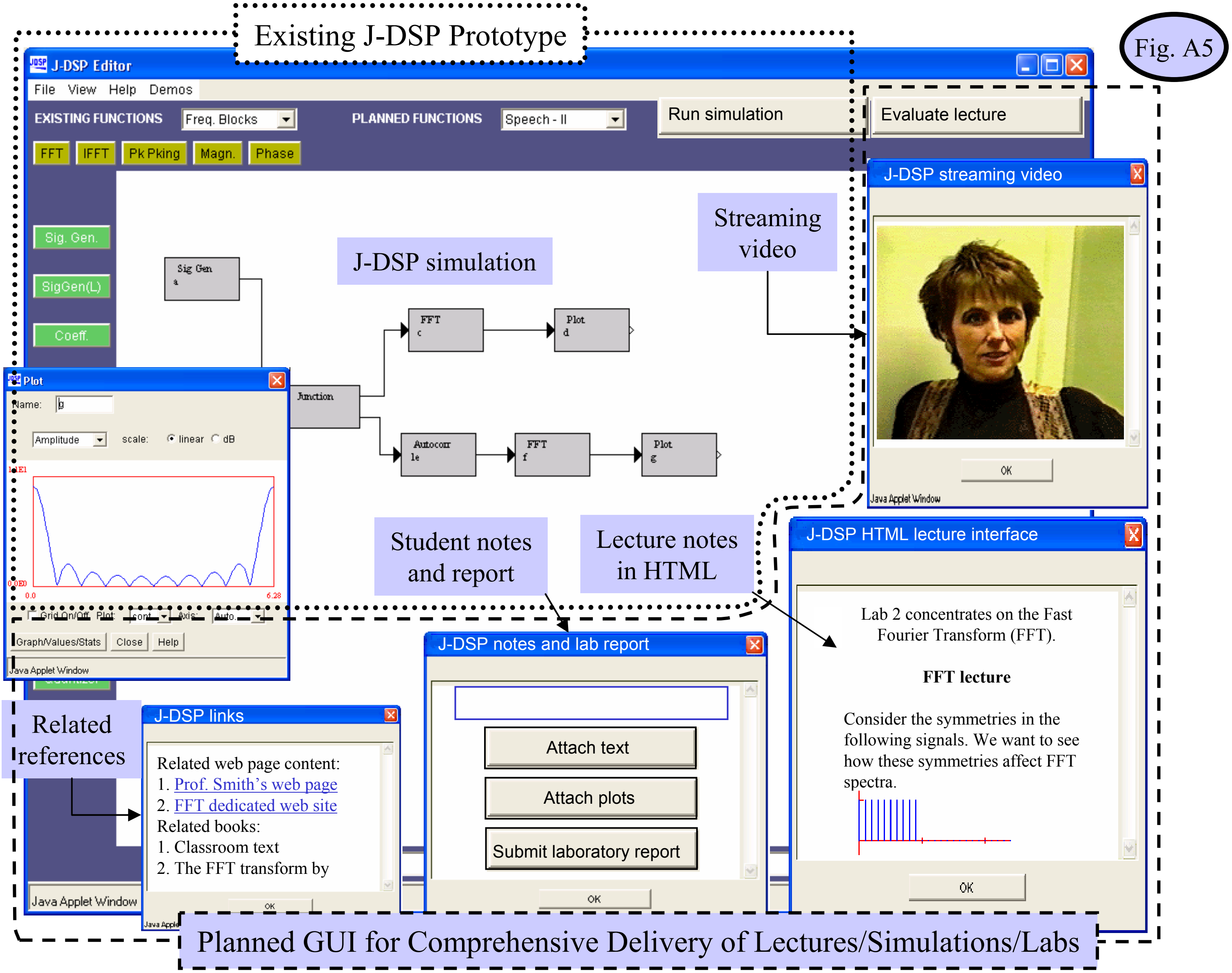
G:\Matlab\work\temp.m

File Edit View Text Debug Breakpoints Web Window Help

```
% MATLAB code generated by J-DSP Editor

clear all; close all;






% Filter (f,6)
NUM6 = [1.0,2.0,-1.0,0.0]
DEN6 = [1.0,1.0,0.0,1.0]
VAR6 = filter(NUM6,DEN6,VAR4);
figure(6), freqz(NUM6,DEN6)


% Plot (b,2)
figure(2), plot(VAR2)
% Plot (h,8)
figure(8), plot(VAR6)
```

1. Prepare demonstration in J-DSP. **2**. Export simulation in MATLAB™ script. **3**. Copy and paste code into MATLAB™.

J-DSP: A full featured educational tool



Existing J-DSP Prototype

Fig. A5

J-DSP simulation

Streaming video

Student notes and report

Lecture notes in HTML

Related references

Planned GUI for Comprehensive Delivery of Lectures/Simulations/Labs

Lab 2 concentrates on the Fast Fourier Transform (FFT).

FFT lecture

Consider the symmetries in the following signals. We want to see how these symmetries affect FFT spectra.

Related web page content:
1. Prof. Smith's web page
2. FFT dedicated web site
Related books:
1. Classroom text
2. The FFT transform by

Laboratory specific evaluation questions

This is a list of laboratory specific evaluation questions students were required to complete after finishing an assigned laboratory exercise with J-DSP.

Lab 2: The Z-transform and Frequency responses

Q1. **The contents of this exercise improved your understanding of the concepts of the Z transform**

○ Strongly Agree
○ Agree
○ Neutral
○ Disagree
○ Strongly Disagree

Q2. **Performing this exercise, you learned how to generate a sinusoid with a digital filter.**

○ Strongly Agree
○ Agree
○ Neutral
○ Disagree
○ Strongly Disagree

Q3. **Can you now understand more clearly the relationship of the impulse response with the transfer function?**

○ Yes
○ No

Q4. **Did you experience any problems in terms of connection, time to implement etc? Please describe.**

Q5. **Was there enough information in the help screens to assist you in using the blocks?**

- ○ Yes
- ○ No
- ○ Did not use the help screens

Q6. **Is there any particular concept, which you could not grasp earlier from your text book/class, but became clear from this exercise? If so, please describe.**

Q7. **Performing the exercises, you are now more comfortable with these topics.**

- ○ Yes
- ○ No

Q8. **Setting up the required lab simulations was pretty easy.**

- ○ Strongly Agree
- ○ Agree
- ○ Neutral
- ○ Disagree
- ○ Strongly Disagree

Q9. **Can you suggest an exercise along the lines of this one?**

Q10. **Please suggest possible improvements relative to this lab (such as redesigning of dialog box of a block - what to add or delete etc).**

Lab 3: Frequency responses and pole-zero plots

Q1. **The contents of this exercise improved your understanding of the concepts of pole-zero plots and frequency responses.**

- ○ Strongly Agree
- ○ Agree
- ○ Neutral
- ○ Disagree
- ○ Strongly Disagree

Q2. **You now understand more clearly the relationship of the location of poles and zeros with the frequency response of a system.**

- ○ Strongly Agree
- ○ Agree
- ○ Neutral
- ○ Disagree
- ○ Strongly Disagree

Q3. **What is the difference between PZ Placement and PZ-Plot block?**

Q4. **Did you add the poles (or/and zeros) graphically (or manually) in PZ-Placement block? Which one do you prefer? Why?**

Q5. **Did you experience any problems in terms of connection, time to implement etc? Please describe.**

Q6. **Was there enough information in the help screens to assist you in using the blocks?**

○ Yes

○ No

○ Did not use the help screens

Q7. **Is there any particular concept, which you could not grasp earlier from your text book/class, but became clear from this exercise? If so, please describe.**

Q8. **Performing the exercises, you are now more comfortable with these topics.**

○ Yes

○ No

Q9. **Setting up the required lab simulations was pretty easy.**

- ○ Strongly Agree
- ○ Agree
- ○ Neutral
- ○ Disagree
- ○ Strongly Disagree

Q10. **Can you suggest an exercise along the lines of this one?**

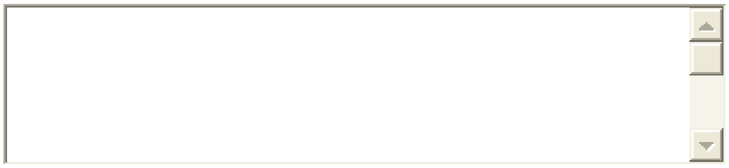
Q11. **Please suggest possible improvements relative to this lab (such as redesigning of dialog box of a block - what to add or delete etc).**

Lab 4: FIR and IIR filters

Q1. **The contents of this exercise helped you understand the concepts of FIR and IIR filter design.**

- ○ Strongly Agree
- ○ Agree
- ○ Neutral
- ○ Disagree
- ○ Strongly Disagree

Q2. **Which part of the exercise helped you the most to understand the concepts of FIR and IIR filter design?**



Q3. **Performing the exercise, can you decide which window to use for sharp transitions in a filter?**

- ○ Strongly Agree
- ○ Agree
- ○ Neutral
- ○ Disagree
- ○ Strongly Disagree

Q4. **Through the IIR filter exercise, you know which IIR filters give a monotonic pass band.**

- ○ Strongly Agree
- ○ Agree
- ○ Neutral
- ○ Disagree
- ○ Strongly Disagree

Q5. **Did you experience any problems in terms of connection, time to implement etc? Please describe.**

Q6. **Was there enough information in the help screens to assist you in using the blocks?**

○ Yes

○ No

○ Did not use the help screens

Q7. **Is there any particular concept, which you could not grasp earlier from your text book/class, but became clear from this exercise? If so, please describe.**

Q8. **Performing the exercises, you are now more comfortable with these topics.**

○ Yes

○ No

Q9. **Setting up the required lab simulations was pretty easy.**

○ Strongly Agree

○ Agree

○ Neutral

○ Disagree

○ Strongly Disagree

Q10. **Can you suggest an exercise along the lines of this one?**

Q11. **Please suggest possible improvements relative to this lab (such as redesigning of dialog box of a block - what to add or delete etc).**

Lab 5: The Fast Fourier Transform (FFT)

Q1. **The contents of this exercise helped you understand the general concepts of using Fast Fourier transform in signal analysis.**

○ Strongly Agree

○ Agree

○ Neutral

○ Disagree

○ Strongly Disagree

Q2. **Which part of the exercise helped you the most to understand how and when to use the FFT?**

Q3. **The exercise helped you to clearly visualize signal symmetries on the FFT spectra.**

○ Strongly Agree

○ Agree

○ Neutral

○ Disagree

○ Strongly Disagree

Q4. **This lab strengthened your perception about various window trade-offs.**

- ○ Strongly Agree
- ○ Agree
- ○ Neutral
- ○ Disagree
- ○ Strongly Disagree

Q5. **By performing this exercise, you understand that spectral resolution of the FFT is limited by frame size, window type and window size.**

- ○ Yes
- ○ No

Q6. **Did you experience any problems in terms of connection, time to implement etc? Please describe.**

Q7. **Was there enough information in the help screens to assist you in using the blocks?**

- ○ Yes
- ○ No
- ○ Did not use the help screens

Q8. **Is there any particular concept, which you could not grasp earlier from your text book/class, but became clear from this exercise? If so, please describe.**

Q9. **Performing the exercises, you are now more comfortable with these topics.**

○ Yes

○ No

Q10. **Setting up the required lab simulations was pretty easy.**

○ Strongly Agree

○ Agree

○ Neutral

○ Disagree

○ Strongly Disagree

Q11. **Can you suggest an exercise along the lines of this one?**

Q12. **Please suggest possible improvements relative to this lab (such as redesigning of dialog box of a block - what to add or delete etc).**

General evaluation questions

This is a list of the general evaluation questions students were required to complete after using J-DSP. This list is also available online at http://jdsp.asu.edu for all other users.

Part 1 of 3

Q1. **What type of Internet access did you use?**

- □ 28.8/56.6 modem
- □ DSL/cable modem
- □ LAN

Q2. **How long did it take to get familiar with the basics of the J-DSP environment?**

- □ 15 minutes or less
- □ Half an hour
- □ An hour
- □ More than an hour

Q3. **How would you rate the JDSP concept on a scale of 1 (bad) to 10 (excellent)?**

Q4. **Did the demos help you understand the J-DSP environment?**

- □ Yes
- □ No
- □ Did not view the demos at all

Q5. **Establishing and connecting blocks is easy.**

- □ Strongly Agree
- □ Agree
- □ Neutral

○ Disagree

○ Strongly Disagree

Q6. **Do you like the idea of an Internet based simulation tool such as J-DSP?**

○ Yes

○ No

Q7. **Overall, do you think its worthwhile using J-DSP as opposed to MATLAB^{TM} for small tasks?**

○ Yes

○ No

Q8. **Your occupation:**

○ High School student

○ Undergraduate student

○ Graduate Student

○ Faculty/Instructor

○ Researcher

○ DSP practitioner

○ Technician

○ Engineering manager

○ DSP hobbyist

○ Other

Q9. **If you are a student or faculty, please give the name of your university.**

Q10. **Which continent are you logging in from?**

○ North America

□ South America

□ Europe

□ Asia

□ Australia

□ Africa

Q11. **In your opinion, is this type of on-line lab concept beneficial for distance learning?**

□ Yes

□ No

Q12. **Did you experience any problems while starting the J-DSP editor?**

□ Yes

□ No

Q13. **Any comments?**

Part 2 of 3

Q1. **The J-DSP menus and blocks are organized in an intuitive and user-friendly manner.**

□ Strongly Agree

□ Agree

□ Neutral

□ Disagree

□ Strongly Disagree

Q2. **Changing the options in the blocks is easy and convenient.**

□ Strongly Agree

□ Agree

- □ Neutral
- □ Disagree
- □ Strongly Disagree

Q3. **The graphical interface of J-DSP is intuitive and user-friendly.**

- □ Strongly Agree
- □ Agree
- □ Neutral
- □ Disagree
- □ Strongly Disagree

Q4. **Did you find it easy to define a signal using the Sig.Gen block? Do you suggest any other options?**

Q5. **Were the options in the Plot block sufficient to clearly visualize the output?**

- □ Yes
- □ No

Q6. **When the Plot block was used for a frequency-domain graph, were you able to zoom and identify spectral peaks?**

- □ Yes
- □ No

Q7. **Are the help screens adequate?**

- □ Yes
- □ No

Q8. **Would you consider using J-DSP for small simulations apart from the lab exercises?**

□ Yes

□ No

Q9. **Do you think with the help of a simple manual, it is possible to learn the basics of DSP by using J-DSP?**

□ Yes

□ No

Q10. **Should the J-DSP editor be established as a full-fledged tool?**

□ Yes

□ No

Part 3 of 3

Q1. **What would you change in the J-DSP program to make it more useful and user-friendly?**

Q2. **Please give your views for improvement of the help screens.**

Q3. **Was the process of entering filter coefficients intuitive? Any suggestions?**

Q4. **Do you suggest any other signal types for the Sig.Gen block?**

Q5. **Please describe any errors or bugs you encountered.**

LAB SPECIFIC ASSESSMENT

CONTENTS

ASSESSMENT OF THE JAVA-DSP (J-DSP) ON-LINE LABORATORY SOFTWARE

This report presents assessment results of the Java-DSP (J-DSP) on-line laboratory. J-DSP software has been developed from the ground up at Arizona State University (ASU) to support the computer lab portion of the senior-level DSP course EEE407. The software enables on-line interactive DSP laboratories. Along with the software, we have developed several J-DSP laboratory exercises that have been posted on the internet. Assessment of the EEE 407 labs was carried both on the web and as part of the instructor and class evaluation. The web-based assessments have been organized into: general software assessments, general laboratory assessments, concept-specific lab-by-lab assessments, and differential pre/post assessment for each lab. Statistical and qualitative evaluations have been compiled for all the J-DSP laboratories and are described in the rest of the report.

Index Terms – Assessment of J-DSP, On-line labs, filter design, ASU EEE407 DSP Course

1. INTRODUCTION

Java-DSP (J-DSP) is an NSF funded on-line software environment that was developed to provide on-line laboratory experiences to distance learning and on-campus students. J-DSP consists of object-oriented Java software that resides on the internet and enables students to build simple and complex simulations of DSP algorithms. The core software environment was initially developed in the late 1990s [1]. Since then the J-DSP concept has been continuously developed and updated with a series of new functions and on-line laboratories [2] as well as other modular web content. The software resides on several servers and is used by students taking the DSP class at ASU and in other universities where beta sites have been established. Through the years, the software has been thoroughly verified and formal dissemination occurred at the 2002 FIE Conference and at the 2002 IEEE DSP Education workshop in Atlanta. The software has been disseminated to more than 50 instructors throughout the world.

Although initially, J-DSP assessment was carried within the course evaluation forms, in the last year we assessed separately the degree of learning attributed specifically to J-DSP. We focused in particular on assessing whether J-DSP accelerated the learning curve in the DSP class. Both on-line and off-line materials have been developed. Two types of on-line forms have been developed i.e., general and concept-specific. In the general forms, the students are asked to provide general qualitative and quantitative evaluation of the J-DSP concept including logistics, accessibility, convenience, demographics, academic standing, etc. In concept-specific forms, students evaluated each laboratory task with regard to its impact on learning specific DSP concepts. Our newest assessment instruments assess learning attributed specifically to J-DSP by testing DSP concepts before (pre-assessment) the J-DSP lab and after (post-assessment) the J-DSP lab. In this report, we present detailed qualitative and statistical results of this comprehensive assessment effort and describe how J-DSP contributes to the learning of several key DSP concepts.

2. DSP LAB ASSIGNMENTS

The laboratory assignments of the EEE407 DSP course are designed to enhance student learning in the DSP class. Several hands-on J-DSP computer exercises have been carefully developed not only to reinforce the DSP concepts covered in class but also expose students to complimentary material that is not covered in detail neither in class nor in text books. The computational nature of these laboratories and the inclusion of real-life signals made J-DSP particularly useful in providing engineering intuition and valuable hands-on experiences. On the other hand, simulations of quantization effects and manipulations of truncated signals made students aware of the limitations associated with processing real-life signals with DSP algorithms. Currently EEE407 includes six computer lab assignments that are assigned on a weekly basis.

1) Difference equations and the Z-Transform,

2) Pole-Zero Plots and Frequency Responses,

3) FIR and IIR Filter Design,

4) The Fast Fourier Transform (FFT),

5) Multi-rate Signal Processing and QMF banks,

6) Introduction to Random Signal Processing.

All these labs require on-line access to the J-DSP editor. Each laboratory contains several problems and exercises. The students have to complete an on-line quiz and submit graphs and comments in the form of an electronic report

3. DESCRIPTION OF THE LABS

Lab 1: Difference Equations and the Z-Transform

The objective of this lab is to introduce the students to the concepts of linear-time-invariant (LTI) systems, z-transforms, and the impulse response of the LTI systems. Moreover, the students observe the filtering effects, and get familiarized with the source-filter configuration. Six problems have been developed for this lab. In problem 1, students are asked to simulate a digital filter using a given transfer function. Figure 1 shows an example simulation performed using J-DSP. In problem 2, the students are asked to design a digital oscillator. In problem 3, a finite impulse response (FIR) filter is provided and the students are asked to observe the behavior of the system for different inputs. In problem 4, the students study symmetric impulse responses. In problem 5, the students compute the transfer function for various pole-zero (PZ) representations. In problem 6, they simulate cascade- and parallel-configurations.

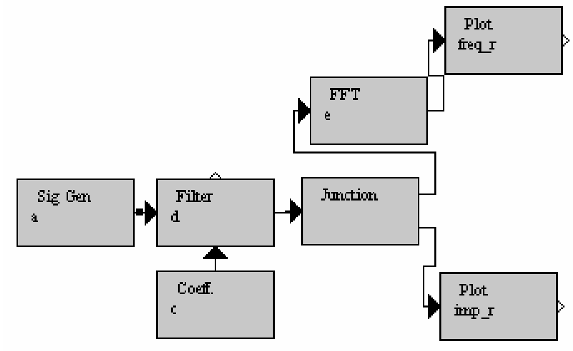


FIGURE 1

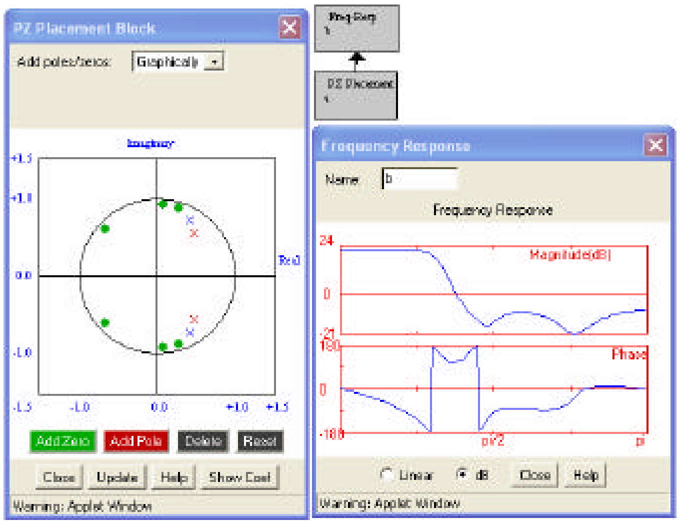
J-DSP BLOCKS USED IN SIMULATION OF DIGITAL FILTER IN LAB1, PROB 1

Lab2: Pole-Zero Plots and Frequency Responses

This lab deals with the effect of pole and zero locations on the magnitude frequency response. First, the relationship between the pole-zero plot and the magnitude response of a system is covered. Four problems are assigned in this lab. In Problem 1, the students are asked to find the poles and zeros and observe the frequency response of a given filter. In Problem 2, the students observe the variations in the frequency response by graphically moving the poles and zeros in the z-domain or, by manually entering their values. In Problem 3, low-pass and high-pass filters are designed based on the pole-zero (PZ) placement method. Figure 2 shows the design of a low-pass filter using PZ placement. In Problem 4, the pole-zero locations and the frequency response for an all-pass filter is examined.

Lab 3: FIR and IIR Filter Design

This exercise examines the four types of symmetric impulse responses that result in linear-phase. In addition, the constraints on the zeros of linear-phase filters are studied. FIR filter design using the Fourier series and tapered windows are covered. Seven problems are assigned in this lab. Problem 1 involves the design of FIR filters, i.e., the students are asked to observe the frequency response, Z-domain symmetry, and the group delay for these filters. Problem 2 deals with the design of low-pass filters by truncating the ideal impulse response using windows. The following window types are supported in J-DSP: rectangular, Bartlett, Hamming, Hanning, and Kaiser.



Add poles/zeros: Graphically

Name b

FIGURE 2

DESIGN OF LOW PASS FILTER USING PZ PLACMENT BLOCK IN LAB 2, PROB 3

In problem 3, the students are asked to design high-pass filters using the Kaiser window. Problem 4 deals with FIR filter design using frequency-sampling.method In problem 5, the students are asked to design optimal FIR filters using the Parks-McClellan algorithm. In problem 6, students are asked to compare the sidelobe levels obtained for the filters designed using the Parks-McClellan method, the Kaiser design, and the frequency-sampling method. Problem 7 deals with the design of IIR filters using bi-linear analog filter approximations. In particular, students are asked to design and compare Butterworth, Chebychev I, Chebychev II, and Elliptic digital filters.

Lab 4: The Fast Fourier Transform (FFT)

In this lab, students learn various concepts related to the use of the DFT and the FFT. In particular, in this lab, students gain familiarity with the estimation of DFT spectra, DFT spectral leakage, DFT resolution, the Parseval's theorem for the DFT, FFT properties and symmetries, and signal estimation and reconstruction using the FFT. In Problem 1, students examine symmetries of the FFT. In Problem 2, students observe the effect of zero-padding and windowing on the FFT spectra. The blocks used in J-DSP for Problem 2 are shown in Figure 3. In Problem 3, the students are asked to examine and compare the FFT spectra of various signals. In Problem 4, students are provided with two sinusoids that are closely spaced in the frequency, and are asked to examine the FFT spectra with several windows

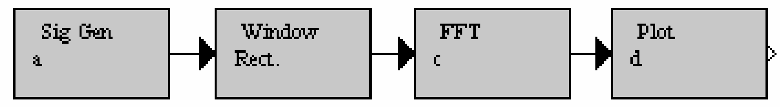


FIGURE 3

BLOCKS USED IN FIR FILTER DESIGN BY WINDOWING IN LAB4, PROB 2

Lab 5: Multi-rate Signal Processing and QMF banks

The goal of this exercise is to examine the effects and the use of the sampling rate conversion and simulate a two-band quadrature mirror filter (QMF) bank. The students get familiar with the up-sampling and down-sampling rules. They also study the effects of aliasing and imperfect reconstruction in decimation and interpolation of digital signals. Four problems are assigned in this lab. In Problem 1, students examine the effect of down-sampling and up-sampling on FFT spectra. In Problem 2, students design a fractional sampler. In Problems 3 and 4, students implement and evaluate a two-band QMF bank and a tree structured QMF using J-DSP.

Lab 6: Introduction to Random Signal Processing

This lab is optional and covers elements of spectral analysis of random signals. The goal of this exercise is to provide students with the basics of classical and parametric spectral estimation. J-DSP has functions for estimating periodograms and correlograms. It can also estimate parametric autoregressive (AR) spectra by using the linear predictive coding (LPC) functions. Correlograms are established by connecting the output of the long signal generator to the autocorrelation block. This is followed by a connection to a lag window and then to SymCorr and subsequently to the correlogram block. The graph panel will then show the correlogram. The length of data, the window, and the length of correlation are selectable and enables students to experiment with trade-offs of spectral resolution and statistical variance. A task to estimate the spectrum of stationary data using a periodogram and a correlogram is assigned. The performance characteristics of the two estimators are evaluated in terms of variance and resolution capability. This process is repeated for the AR spectral estimator.

4. LAB SUBMISSION PROCEDURE

All the Lab assignments are accessed on-line and students submit their work on the internet. Each of the students in the EEE407 DSP class is given a user name and password for a lab account. Using lab account, students complete an on-line, media-rich report that includes a quiz that covers the laboratory material. The submitted electronic report contains responses to multiple choice questions, dialog boxes for writing qualitative comments, and facilities to upload the graph files and equations in ‘gif’ format. Upon submitting the report, all of the student’s answers, comments and graphs are placed together in a static HTML that corresponds to each student’s ID. Part of the grading is done automatically while part of it requires instructor intervention. The automatic part is processed by a UNIX shell script on the server computer that grades the answers of the multiple choice and true/false questions.

5. PRE/POST LAB ASSESSMENTS

In addition to the general assessment of J-DSP and the pertinent exercises carried in EEE407 in the Fall of 2002, in the spring of 2003 we started running specific pre- and post- lab assessments for each lab. The purpose of the assessments is to survey and evaluate the level of student's understanding of the key DSP concepts before and after performing a particular J-DSP lab assignment. Thus the statistics obtained from these assessments give us feedback on how the J-DSP lab assignments helped the students in learning the key concepts on a particular topic. The pre/post quiz and the lab are assigned after the relevant theory has been introduced in class. We did this in order to ensure that all the students have had some, or ideally the same exposure to the topics covered in the lab, so that we can isolate specifically the effect of J-DSP labs in their learning. The students are asked to complete the pre-lab assessment before working on that lab. After they complete and submit the lab assignments, they complete the post-lab assessment. The questions on the post-lab assessments are same as the pre-lab assessments but given in a different order.

5.1. Results of Pre/post Lab Assessments

Pre/post assessment results are shown graphically in Figures 4 and 5 for Labs 1 through 5. Lab 6 was optional and by the time this report was submitted we did not have the data available. The assessment results of each of these labs are discussed below.

Lab 1 assessment results

From Figure 6, we see that a total of 6 questions are assigned in Lab 1 assessment. Lab 1 is related to the Z-Transform and the frequency response. In Question 1, the students are asked to determine the impulse response for a given transfer function. We observe that 92% students answer correctly before performing the lab and 92.3% students answer correctly after they have finished the lab. Percentage improvement is negligible as the question was evidently very simple. In Question 2, students were asked to find the poles and zeros of a given transfer function. 88% students answered correctly before they started working on the lab and 92.3% students answered correctly after completing the lab. In Question 3, we asked students to find the impulse response for a given transfer function that is sum of two first-order all-pole filters, i.e., the composite system consists of two parallel systems. 76% students answered correctly before they attempted the J-DSP Lab 2 assignment and 94.9% students answered correctly in the post-lab assignment. The percentage improvement is 18.9% and is noticeable.

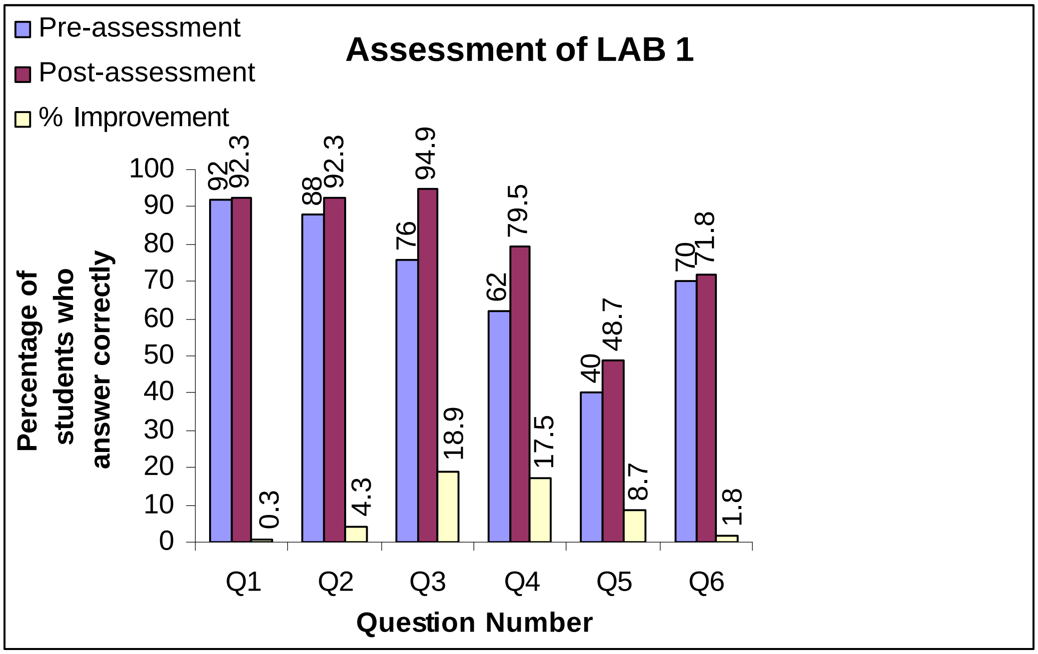


FIGURE 4(A)

THE RESULTS OF LAB 1 ASSESSMENT

In Question 4, a sinusoidal impulse response of a system (i.e. a digital oscillator) is given and the students are asked to choose the true characteristics of the filter out of four choices. 17.5% more students answered correctly after using J-DSP. In Question 5, students are asked if it is possible to suppress completely a sinusoid of a certain frequency using an FIR or an IIR filter. We observed an 8.7% improvement in the post-lab assessment. In Question 6, students are asked about the structure of poles given the coefficients. For this question, only 1.8% improvement was observed. In summary, we observed a 10% average improvement in lab 1 after J-DSP was used.

Lab 2 assessment results

Again we observe in all the questions, students have performed better in the assessment after they have completed their J-DSP labs. Questions 1 and 2 assessed whether students can relate the pole and zero positions to the resulting magnitude frequency response. A 13.1% improvement was realized in question 1 and a 13.29% was realized in question 2 in the post lab assessment. In Question 3, two polynomial functions are given and students are asked if their magnitude responses are equal. A 10.99% improvement is noticed here. In Question 4, students are asked about the filter type (i.e. low-pass/high-pass etc.) of a given transfer function.

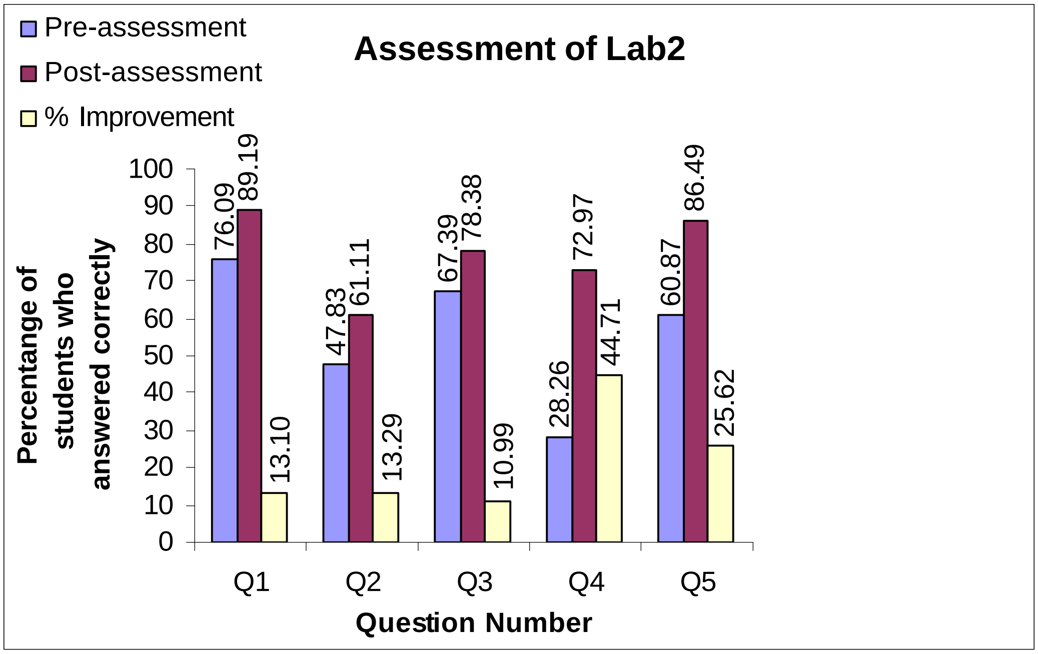


FIGURE 4(B)

THE RESULTS OF LAB 2 ASSESSMENT

We see that only 28.26% of students answered correctly in pre-lab assessment and it is indeed poor but the percentage has increased by 44.71% in the post-lab assessment. This is significant. In Question 5, students are given four choices regarding the change of magnitude response with displacement of the zeros with respect to the unit circle. A noticeable improvement of 25.62% in post-lab is evident.

Lab 3 assessment results

On the average around 11% improvement is evident in each question in the results of pre/post assessment of the lab 3 assignment regarding the design of FIR and IIR filters. In Question 1, the frequency response of an ideal low-pass filter is given and the students are asked to specify whether the filter is FIR or IIR and whether it is causal or non-causal. A 11.9% improvement of the student's performance is evident. Questions 2 and 3 are related to the characteristics of linear-phase FIR filters and 15.46% and 22.37% improvements are observed respectively. These improvements reveal that the lab clarified further the student understanding of linear phase filters. In Question 4, the students are asked to choose an optimal filter design method from four choices, namely, frequency sampling, Parks-McClellan, Kaiser window and Fourier series method. A 12.08% improvement is observed. Question 5 addressed the reasons for which one may choose an FIR filter. Question 6 is set to assess the student's awareness on the constraints posed on the magnitude frequency response by certain linear-phase designs. In Question 7, four impulse responses are given to the students and are asked to choose those that have linear phase. A 19.30% improvement is observed here.

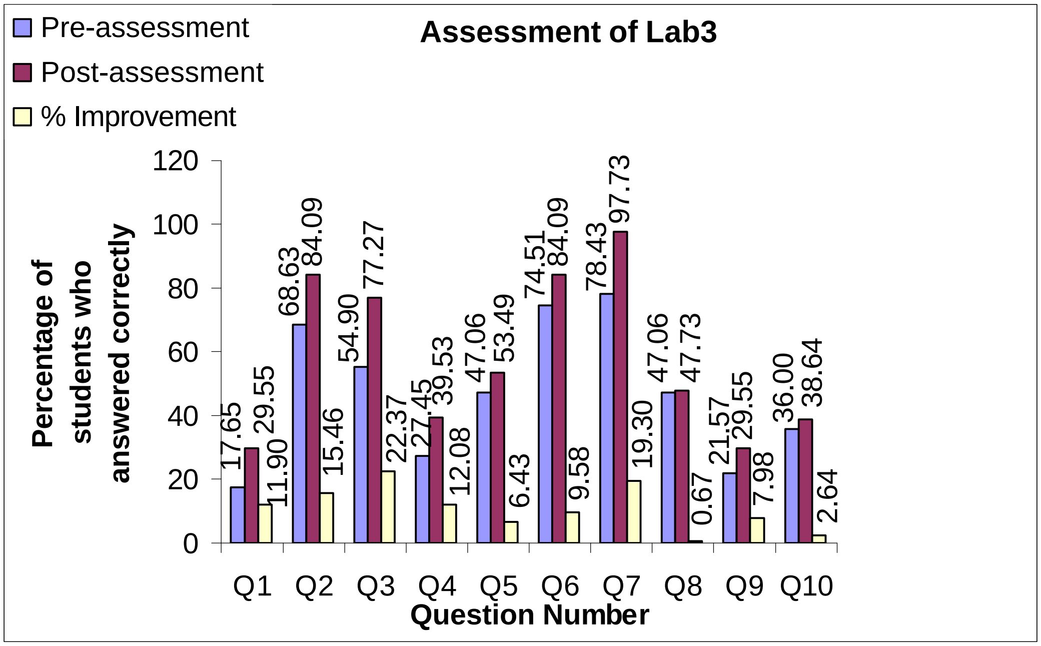


FIGURE 4(C)

THE RESULTS OF LAB 3 ASSESSMENT

Questions 8 and 9 are related to finding the group delay from the phase and impulse responses of FIR filters. Question 10 is set to assess whether students are aware of the fact that FIR design by frequency sampling of ideal frequency responses does not yield an ideal filter.

Lab 4 assessment results

Lab 4 is about the FFT. Question 1 is related to the symmetry properties of the DFT. The improvement observed is 9.47%. Question 2 is regarding the resolution of the DFT. Question 3 was about the mainlobe and sidelobe characteristics of the rectangular window. A 9.23% improvement is observed here. This was a relatively long laboratory exercise and our general impression was that the students were able to understand several concepts on the DFT that are not immediately evident from the lecture and the text book. The students gained valuable experience on spectral resolution and spectral leakage by viewing and interpreting spectra of several bench-mark signals.

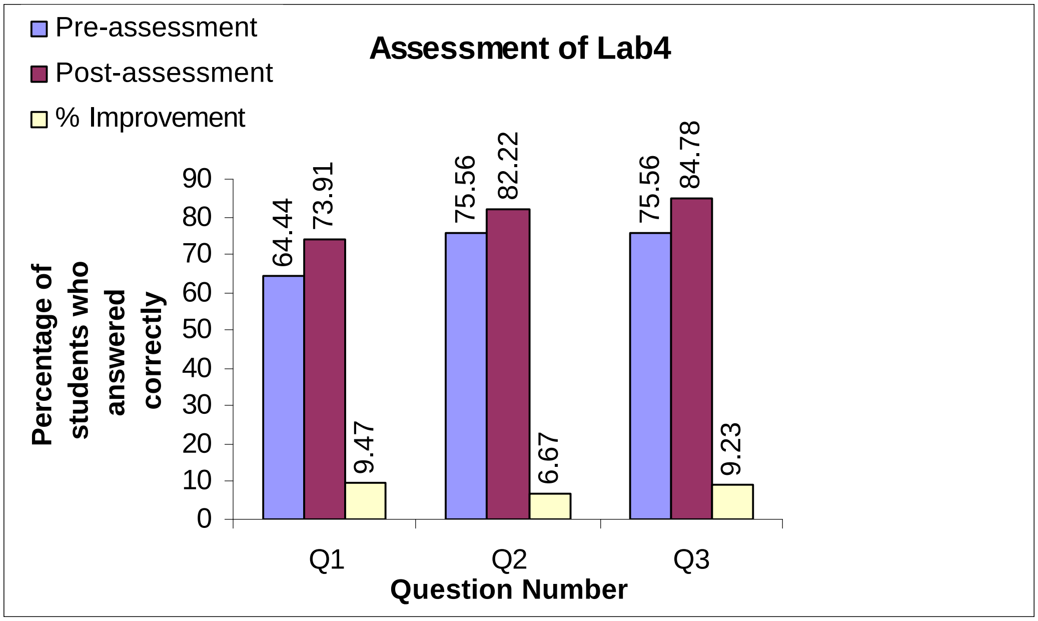


FIGURE 4(D)

THE RESULTS OF LAB 4 ASSESSMENT

Lab 5 assessment results

Lab 5 is on multi-rate signal processing and QMF banks. Figure 5 shows the assessment results of Lab 5. In Question 1, the students are asked to describe how spectral domain signatures are affected by down-sampling and up-sampling.

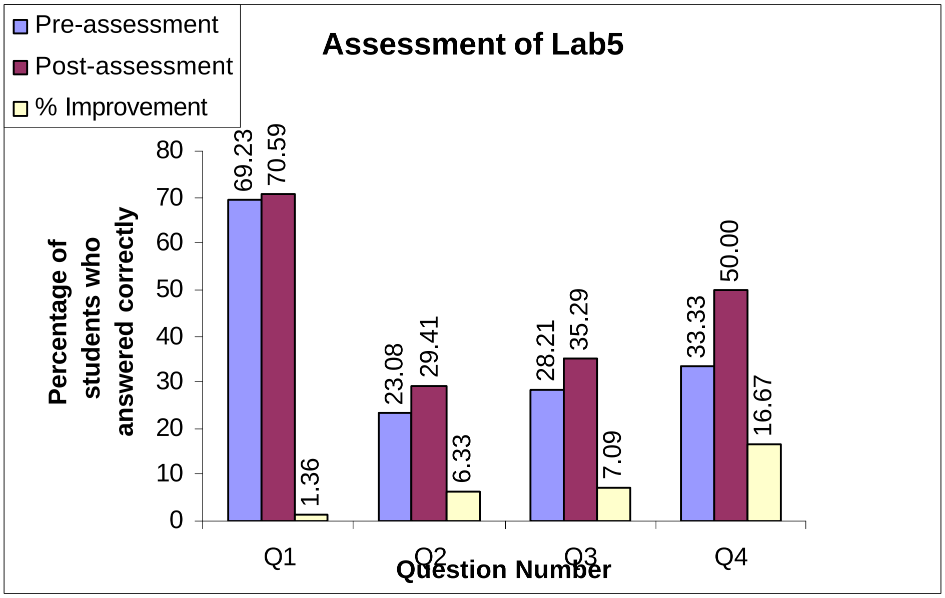


FIGURE 5

THE RESULTS OF LAB5 ASSESSMENT

Question 2 is about the use of a reconstruction filter for interpolation. Question 3 is related with the placement of a quantizer in a QMF bank. An average improvement of 8% is observed in this lab. This laboratory was also one that the students benefited from in that they gained hands-on experience on filter banks which are now common in MP3 players and MPEG video compressors.

6. GENERAL ASSESSMENT

In this assessment, students gave their general subjective opinions on J-DSP and provided us with an idea as to whether the pertinent labs have been useful and helpful to them. In pre/post-lab assessment the questions posed were technical and are set to test the student's level of knowledge before and after performing the J-DSP lab. In the general assessment, the questions are less technical and are posed to get feedback from the students regarding the usefulness of the J-DSP software and associated exercises.

Overall, the responses were very promising. From Table 1 and Figure 6, we see that 95% of the users appreciated the various features of J-DSP as an internet-based simulation tool. From Figure 6, it is clear that it took most (70%) of the users less than half an hour to learn using the software. In fact, 85.5% of the users agreed that they would consider using J-DSP for DSP simulations. Lab specific general assessment was done on Labs 1 through 4 in the Fall 02 semester (the current semester is spring 2003).

TABLE I

STATISTICS BASED ON USER EVALUATIONS OF J-DSP TOOL

Evaluation questions	Strongly Agree (%)	Agree (%)	Neutral (%)	Disagree (%)	Strongly Disagree (%)
1. Establishing and connecting blocks is easy.	53	39	7	1	0
2. The graphical interface of J-DSP is intuitive and user-friendly.	31	63	5	1	0
3. Setting up the required lab simulations was easy	40	52	8	0	0

From Table 2, we see that most of the students (above 90%) agreed that the J-DSP labs helped them understand the DSP related concepts. Also from Table 3, it is evident that more than 90% students are comfortable with the DSP related topics after completing the J-DSP labs. In the evaluation they were also asked if they received additional knowledge by performing the lab assignments in addition to the lectures. A few comments are presented to give a picture of student’s impression. In Lab 1, some of them stated that the lab helped understand better the concept of cascaded and parallel configurations.

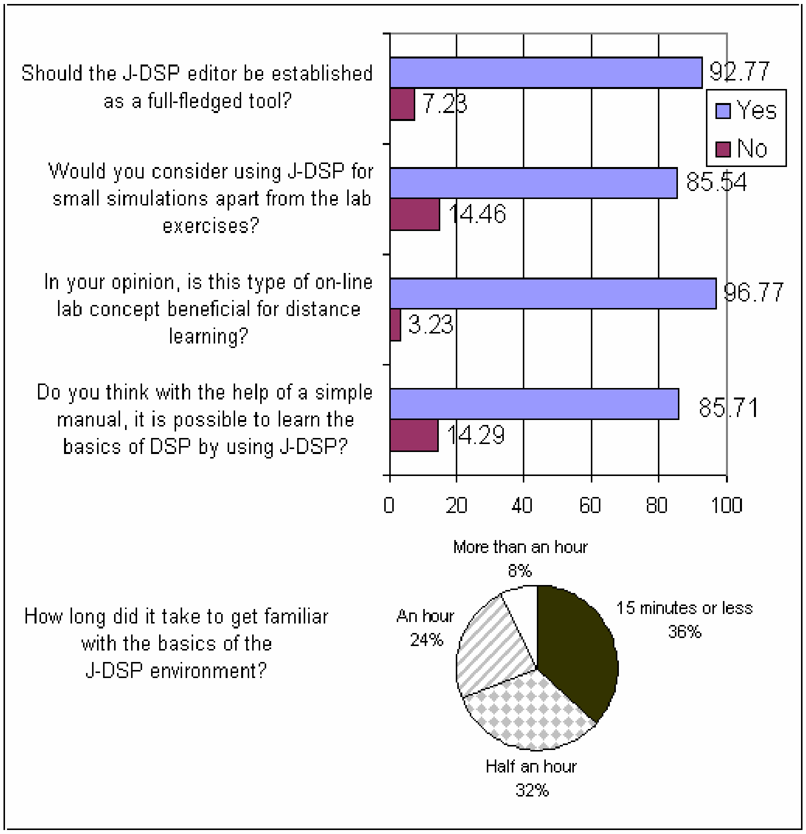


FIGURE 6

USER FEEDBACK REGARDIND J-DSP EDITOR USED IN DSP LAB

Other comments included confirmations from students that J-DSP helped them understand better the issues related to filter design and the fact that pole and zero locations relate to the frequency response.

TABLE II

STATISTICS BASED ON USER EVALUATION OF LABS 1-4

Evaluation questions	Lab No.	Strongly Agree (%)	Agree (%)	Neutral (%)	Disagree (%)	Strongly Disagree (%)

1. Improvement of your understanding of the concepts of the Z- transform	1	50	46	3	1	0
2. Improvement of your understanding of the concepts of pole-zero plots and frequency response	2	47	44	7	1	1
3. Improvement of your understanding of the concepts of FIR and IIR filter design	3	42	47	8	3	0
4. Improvement of your understanding of the general concepts of using FFT in signal analysis.	4	24	61	13	2	0
5. You have learned how to generate a sinusoid with a digital filter	1	29	55	11	3	2
6. You have learned which window to use for sharp transition in a filter from lab 4.	3	42	47	8	3	0
7. The exercise helped you to clearly visualize signal symmetries on the FFT spectra	4	18	65	9	6	2

TABLE III

STATISTICS BASED ON USER EVALUATION (YES/NO) OF LAB 1, 2, 3, AND 4

Evaluation questions	Lab No.	Yes (%)	No (%)
1. Understand more clearly the relationship of the impulse response with the transfer function	1	95	5
2. Understand more clearly that spectral resolution of the FFT is limited by frame size, window type and window size	4	99	1
3. Enough information in the help screen*	1	56	17
	2	53	17
	3	40	23
	4	47	13
4. Performing the exercises, you are now more comfortable with the topics related with each lab assignment	1	93	7
	2	94	6
	3	92	8
	4	90	10

* rest 27%, 30%, 37%, and 40% did not use help screen in Lab 1,2,3 and 4 respectively.

Students also recommended some of the changes in J-DSP on-line tools used in the lab simulations. One common suggestion is to accommodate the facility to save a workspace in J-DSP editor for the future use. This problem is now solved as in the new version of J-DSP there is a facility to import/export the work space as a script file. Also students reported several software bugs that have been fixed. We are especially appreciative of the ability to get immediate feedback from the students on the operation of the software.

7. Conclusions

Different types of assessment instruments have been prepared and disseminated to student users in the ASU EEE 407 class in the Fall 2002 and Spring 2003. Continuous feedback helped correct and improve the J-DSP software and exercises. The students in general found the J-DSP concept very convenient and easy to use. Concept specific assessments and pre/post assessment revealed that several J-DSP functions have been particularly useful in communicating key DSP concepts. J-DSP was proven to be particularly useful in learning issues related with filter design and interpretation of frequency spectra. The J-DSP visualizations involving pole-zero diagrams have shown prominent differences in pre- and post- assessments that lead us to believe that we need to integrate even more animation and develop demonstrations that are dynamic. In all, EEE 407 students asserted that they have benefited from J-DSP and they particularly appreciated the fact that the tool was available on the web from any location. Industry students taking the course from remote sites have been particularly impressed with the tools and exercises and some are using it for routine design and other compact DSP simulations. Here are few useful URLs that can be explored by the interested reader. For accessing all the lab assignments of DSP course use http://www.eas.asu.edu/~eee407. The J-DSP editor can be accessed at http://jdsp.asu.edu. The feedback from the users is also available by following the links on the J-DSP web site.

References

[1] Spanias, A. et al., “Development and Evaluation of a Web-Based Signal and Speech Processing Laboratory for Distance Learning”, *ASEE Computers in Education Journal*, Vol. X, No. 2., April-June 2000, pp. 21-26.

[2] Spanias A. et al, “On-line laboratories for speech and image processing and for communication systems using J-DSP”, *2nd DSP-Education workshop*, Pine Mountain, GA, Oct. 13-16, 2002.

Comments on J-DSP by EEE407 students

obtained from course and lab evaluations

The following is a selection of comments or multiple choice answers made by students who used J-DSP to perform instructor-assigned laboratory exercises. It also includes some answers by J-DSP users from the industry and academia.

B.1. Responses of students to multiple choice questions posted in J-DSP online assessment

Q. "How long did it take to get familiar with the basics of the J-DSP environment?"
A. 15 minutes or less – 62, Half an hour – 53, An hour – 32, More than an hour – 15

Q. "Did the demos help you understand the J-DSP environment?"
A. Yes – 112, No – 14, Did not view the demos at all – 38

Q. "Establishing and connecting blocks is easy."
A. Strongly Agree – 74, Agree – 73, Neutral – 15, Disagree – 1, Strongly Disagree - 0

Q. "Do you like the idea of an Internet based simulation tool such as J-DSP?"
A. Yes – 151, No – 13

Q. "Your occupation:"
A. High School student – 1, Undergraduate student – 60, Graduate Student – 97, Faculty/Instructor – 1, Researcher – 0, DSP practitioner – 1, Technician – 0, Engineering manager – 1, DSP hobbyist – 2, Other - 1

Q. "Which continent are you logging in from?"
A. North America – 151, South America – 0, Europe – 4, Asia – 9, Australia – 0, Africa – 1

Q11. "In your opinion, is this type of on-line lab concept beneficial for distance learning?"
A. Yes – 157, No – 6

Q. "The graphical interface of J-DSP is intuitive and user-friendly."
A. Strongly Agree – 44, Agree – 89, Neutral – 11, Disagree – 1, Strongly Disagree – 1

Q. "Would you consider using J-DSP for small simulations apart from the lab exercises?"
A. Yes – 128, No – 17

Q9. "Do you think with the help of a simple manual, it is possible to learn the basics of DSP by using J-DSP?"
A. Yes – 127, No – 19

Q10. "Should the J-DSP editor be established as a full-fledged tool?"
A. Yes – 132, No – 14

B.2. Comments by students obtained from J-DSP online assessment

"excellent simulation of DSP practical work with high level of simplicity and flexibility."

"bravo , good product"

"More functions, such like save/load, cut/paste."

"Very Good Program, I think it would be nice if we could save a block diagram along with coefficients etc and get back to it later and work"

"Nice concept!"

"This is very easy to learn for beginners"

"A great concept that should be used in all areas of study."

"It is a great concept and it has been programmed very well."

"This is a good program to study DSP."

"It's totally good."

"Overall I think it's quite an intuitive and interesting tool which gives the student a real feel of the subject."

"This is good, but will be better when more capabilities are built in such as being able to save diagrams and hear sound better and longer."

This is a good tool. However a better help menu will be welcomed"

Participating University and Industry Test Sites

As part of our dissemination and testing activities, we have already contacted and obtained preliminary agreements from faculty in several undergraduate programs as well as engineers in relevant fields in the industry. Universities that have already agreed to evaluate the modules and all relevant materials include:

- Georgia Tech
- University of Southern California
- University of Maryland
- University of Minnesota
- University of New Mexico
- University of Texas-Austin
- University of Texas-Dallas
- University of Central Florida
- Northeastern University

We also have agreements with several other universities that have previously obtained the J-DSP software and agreed to evaluate future updates. Universities that agreed to establish J-DSP web sites and test lab content include:

- University of Kent
- Marquette University
- Stevens Institute of Technology
- Georgia Institute of Technology
- Blekinge Institute of Technology
- Drexel University
- University of Nebraska
- Cal Poly Pomona
- University of Detroit-Mercy
- University of Pennsylvania
- Iowa State University
- University of New Brunswick
- University of Alabama-Huntsville
- Rice University
- Massachusetts Institute of Technology
- University of Akron
- University of Connecticut
- University of Puerto Rico
- Clemson University
- North Carolina State University
- Ecole Nationale
- Polytechnique-Algeria
- Bogazici University-Turkey
- University of Rhode Island

Industry test sites include:

- Texas Instruments
- Motorola
- Intel
- Semy Engineering
- General Dynamics
- Honeywell
- Nokia

We also add that we have a commitment from a DARPA program manager to disseminate our materials to relevant federal entities.

J-DSP dissemination efforts

This is a list of people who obtained the J-DSP software for evaluation and/or use at their own academic institutions.

NAME	ORGANIZATION
NABIL ZAKRIA	UNIVERSITY OF KENT
MICHAEL JOHNSON	MARQUETTE UNIVERSITY
A ELLCONY	KENT UNIVERSITY
HONG MAN	STEVENS INSTITUTE OF TECHNOLOGY
DAVID ANDERSON	GEORGIA INSTITUTE OF TECHNOLOGY
LARS-OLOF LARSON	BLEKINGE INSTITUTE OF TECHNOLOGY
JAN MARK DE HAAN	BLEKINGE INSTITUTE OF TECHNOLOGY
RZHANG	DREXEL UNIVERSITY
E. N. BIDEN	UNIVERSITY OF NEW BRUNSWICK, CANADA
ALAN FELZER	CAL POLY POMONA
SHUVRA DAS	UNIVERSITY OF DETROIT MERCY
MITCHELL LITT	UNIVERSITY OF PENNSYLVANIA
JOHN PROAKIS	NORTHEASTERN UNIVERSITY
JULIE DICKERSON	IOWA STATE UNIVERSITY
C. S. BURRUS	RICE UNIVERSITY
ANTHONY BESSIOS	TEXAS INSTRUMENTS
JULIE GREEN BERG	MASSACHUSETTS INSTITUTE OF TECHNOLOGY
N. MOHANKML	UNIVERSITY OF DETROIT MERCY
DALE MUGLER	UNIVERSITY OF AKRON
ERIC SOULSBY	UNIVERSITY OF CONNECTICUT
DOMINGO RODRIGUEZ	UNIVERSITY OF PUERTO RICO
PRADIP SRIMANI	CLEMSON UNIVERSITY
JOEL TRUSSELL	NORTH CAROLINA STATE UNIVERSITY
FATIHA MERAZKA	ECOLE NATIONALE POLYTECHNIQUE, ALGERIA
GURDAKUL	BOGAZICI UNIVERSITY, TURKEY
PAUL HASLER	GEORGIA INSTITUTE OF TECHNOLOGY
C. L. MAX NIKIAS	UNIVERSITY OF SOUTHERN CALIFORNIA
MASOUD SALEHI	NORTHEASTERN UNIVERSITY
PHILIP LOIZOU	UNIVERSITY OF TEXAS AT DALLAS
SENNUR ULUKUS	UNIVERSITY OF MARYLAND
TAKIS KASPARIS	UNIVERSITY OF CENTRAL FLORIDA
G. FAYE BOUDREAUX-BARTELS	UNIVERSITY OF RHODE ISLAND
BRIAN L. EVANS	UNIVERSITY OF TEXAS AT AUSTIN
GEORGIOS B. GLANNAKIS	UNIVERSITY OF MINNESOTA
JEFFREY FOUTZ	MOTOROLA
SACHI DASH	HONEYWELL
TED PAINTER	INTEL
GLEN P. ABOUSLEMAN	GENERAL DYNAMICS
SASSAN AHMADI	NOKIA
KEVIN STODDARD	SEMY ENGINEERING
PANOS PAPAMICHALIS	TEXAS INSTRUMENTS

ALEXANDER D. POULARIKAS UNIVERSITY OF ALABAMA IN HUNTSVILLE
YAN WU
JAN E. ODEGARD
JOHN PIERRE
DYLAN DIZON
OSUACDO CLUA
CARLOS GODFRID
M. S. FADALI
MARK BURGE
ALEXEY MATVEYEV
CHRISTINA M. PETERSON

Section M10 J-DSP Scripts

1. J-DSP Script basics

J-DSP is fitted with an interpreter designed to decode parameters contained in a simple Hypertext Markup Language (HTML) file, which when loaded through a browser, invokes the J-DSP editor. The editor in turn interprets the parameters and loads a J-DSP flowgram as described by these parameters. The parameters contained in the HTML file are written in JavaScript and are actually components of a Java™ applet referred here as a J-DSP script. This J-DSP Editor capability has been designed in order to allow instructors save their own J-DSP simulation examples on the Internet, easily integrating interactive content in classroom web sites.

In addition to this introduction, this manual consists of two more main sections that further discuss J-DSP scripts. Section 2 of this manual instructs the user on how to create J-DSP scripts automatically, while section 3 is an elaborate description of how to manually prepare the script code.

Please note that all scripts must be saved in the same directory the J-DSP editor's class files are located in.

2. Generating scripts automatically.

While trivial to prepare manually, the J-DSP Editor has been provided with the ability to automatically generate J-DSP scripts. A user simply needs to create the desired flowgram using the familiar drag and drop procedure of the editor. Then, by selecting "*File"* and "*Export as Script"*, the user obtains the script, ready to use in an HTML file. This automatically generated script also includes all the blocks parameters, exactly as they where defined when the simulation's flowgram was saved, providing full control over a saved simulation. Section 2.1 contains step by step instructions on how to create your own J-DSP scripts.

2.1. Scripts in a flash.

A few steps is all that is needed to get a J-DSP script and save it over the Internet for others to use.

Step1: Start the J-DSP Editor and create a flowgram as desired. Don't forget to set the desired parameters to all the parts.

Step 2: From the main menu, select "*File"* and then "*Export as Script"* as shown in figure 1. A new window appears that contains the code describing the simulation, as shown in figure 2. This will be referred to as the J-DSP script window. Note that by default, the window presents only the applet code which can be placed inside the body of an existing HTML file. If the entire HTML file is needed, select "*Applet in HTML code"* from the drop down box of the script window.

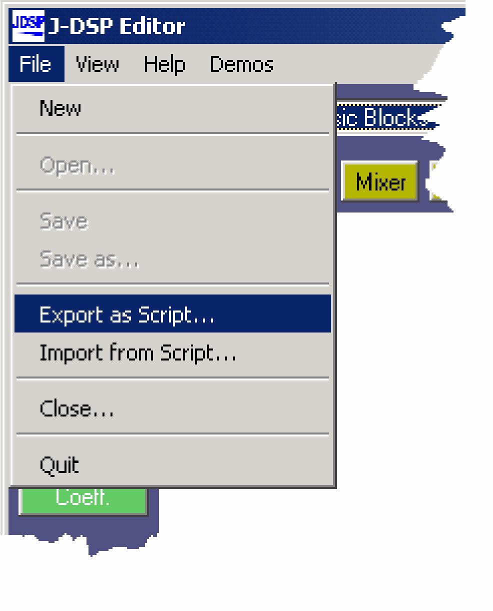
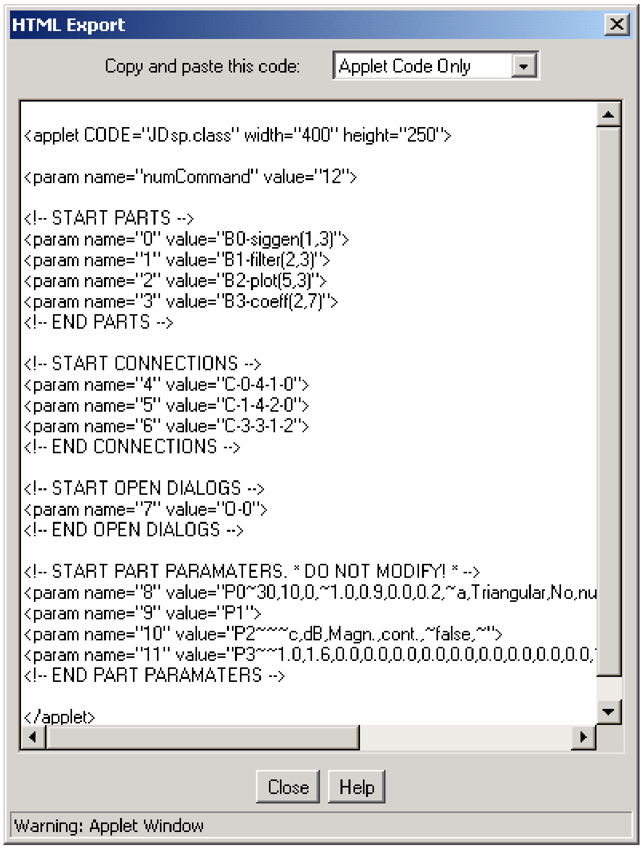


Figure 1: Selecting *File* and then *Export as Script*



```
<applet CODE="JDsp.class" width="400" height="250">

<param name="numCommand" value="12">

<!-- START PARTS -->
<param name="0" value="B0-siggen(1,3)">
<param name="1" value="B1-filter(2,3)">
<param name="2" value="B2-plot(5,3)">
<param name="3" value="B3-coeff(2,7)">
<!-- END PARTS -->

<!-- START CONNECTIONS -->
<param name="4" value="C-0-4-1-0">
<param name="5" value="C-1-4-2-0">
<param name="6" value="C-3-3-1-2">
<!-- END CONNECTIONS -->

<!-- START OPEN DIALOGS -->
<param name="7" value="0-0">
<!-- END OPEN DIALOGS -->

<!-- START PART PARAMATERS. * DO NOT MODIFY! * -->
<param name="8" value="P0~30,10,0,~1.0,0.9,0.0,0.2,~a,Triangular,No,nu
<param name="9" value="P1">
<param name="10" value="P2~~~c,dB,Magn.,cont.,~false,~">
<param name="11" value="P3~~1.0,1.6,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
<!-- END PART PARAMATERS -->

</applet>
```

Figure 2: J-DSP script window

Step 3: Using the mouse, select the code from the window and press *Ctrl-C* to copy it into the clipboard. Some users may use a right click and then *Copy*, depending on the Java version used on their computer. Non-Windows™ users should be able to follow a similar procedure. ***Note***: If you are not able to use Ctrl-C to copy, please install the latest java virtual machine from Sun found at www.java.sun/getjava/. Instructions are provided in the troubleshooting section of our web site located at http://jdsp.asu.edu.

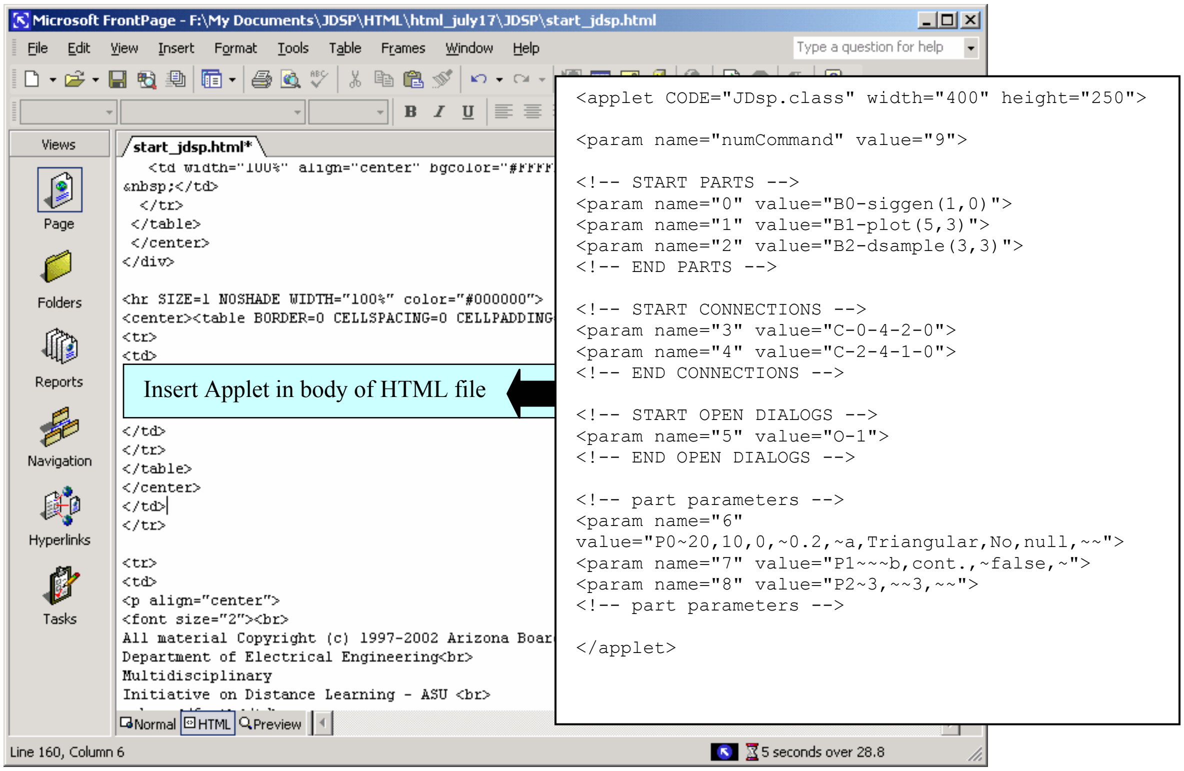
Step 4: If you wish to place the applet in an existing HTML file, simply paste the script code at the location where you desire the J-DSP Editor applet to appear. If you do not have an HTML file, copy the entire HTML code into a text editor and save the file with an .html or .htm extension. Steps 4.1 and 4.2 provide some extra details and can be skipped if you feel comfortable with what was just mentioned.

Step 4.1: Saving into an existing HTML file:

a. Use an HTML editor or any text editor capable of reading ASCII files to load the *filename.html* or *filename.htm* file you wish to use for the script. If your HTML editor is a “what you see is what you get” (WYSIWYG) editor, make sure to select to edit the HTML code itself.

b. Place the cursor where desired and paste the applet code using *Ctrl-V* or right click and *Paste* as shown in figure 3.

c. Save the file and then load it in a browser: ***Note***: Do not attempt to view the HTML file through the HTML editor’s internal browser, as it might not be a fully functional browser and therefore not capable of displaying active content like Java™ applets.



```
   <td width="100%" align="center" bgcolor="#FFFF
&nbsp;</td>
  </tr>
 </table>
 </center>
</div>

<hr SIZE=1 NOSHADE WIDTH="100%" color="#000000">
<center><table BORDER=0 CELLSPACING=0 CELLPADDING
<tr>
<td>
```

Insert Applet in body of HTML file

```
</td>
</tr>
</table>
</center>
</td>
</tr>

<tr>
<td>
<p align="center">
<font size="2"><br>
All material Copyright (c) 1997-2002 Arizona Boar
Department of Electrical Engineering<br>
Multidisciplinary
Initiative on Distance Learning - ASU <br>
```

```
<applet CODE="JDsp.class" width="400" height="250">

<param name="numCommand" value="9">

<!-- START PARTS -->
<param name="0" value="B0-siggen(1,0)">
<param name="1" value="B1-plot(5,3)">
<param name="2" value="B2-dsample(3,3)">
<!-- END PARTS -->

<!-- START CONNECTIONS -->
<param name="3" value="C-0-4-2-0">
<param name="4" value="C-2-4-1-0">
<!-- END CONNECTIONS -->

<!-- START OPEN DIALOGS -->
<param name="5" value="O-1">
<!-- END OPEN DIALOGS -->

<!-- part parameters -->
<param name="6"
value="P0~20,10,0,~0.2,~a,Triangular,No,null,~~">
<param name="7" value="P1~~~b,cont.,~false,~">
<param name="8" value="P2~3,~~3,~~">
<!-- part parameters -->

</applet>
```

Figure 3: Pasting a J-DSP script using Ms FrontPage

Step 4.2: Saving into a new HTML file:

a. Use any text/HTML editor to create a new file.

b. Copy and paste the entire HTML code from the script window into the text/HTML editor.

c. Save the file with an extension of .htm or .html, so that any browser can recognize it. For example, you can name the file: myjdsp.html.

Important note: In every case, make sure that the HTML file containing the script is saved where the J-DSP editor class files are saved, otherwise the script will not run.

Step 5: After saving the HTML file in the same directory as the J-DSP script files, load the file in a browser and press the [Start] button. This should start the J-DSP Editor and load the saved simulation. Make sure you place a link to this HTML file from your web page so that others can have access to it.

You have now created a J-DSP script. If you are interested to learn more on how J-DSP scripts work, you can read further to section 3, otherwise you can safely stop here.

3. Generating scripts manually – Writing the code.

Typically only a few lines of code are necessary to set up and execute a simulation. The following sub-sections describe how to do so, through a simple example.

3.1. The Basics.

An example of html code that establishes and runs a simple J-DSP simulation is listed below:

0	<applet CODE="JDsp.class" width="400" height="250">
1	<param name="numCommand" value="3">
2	<param name="0" value="**B**0-siggen(3,1)">
3	<param name="1" value="**B**1-plot(5,1)">
4	<param name="2" value="**C**-0-4-1-0">
5	</applet>

Note that the J-DSP scripts are placed between the html applet tags, namely, <APPLET> which marks the beginning of the J-DSP applet, and </APPLET> that marks the end of this applet. The applet tag, <APPLET>, also includes the following information: it instructs the browser to load the applet by specifying its sub-class name, width (400 pixels), and height (250 pixels). In our case, the user should always type the following:

<APPLET CODE=”JDsp.class” WIDTH=”400” HEIGHT=”250”>

J-DSP scripts go here

</APPLET>

To create a J-DSP flowgram with scripts, you have to specify a set of parameters that establish the blocks and link them to form a flowgram. These parameters are not associated with the DSP simulations but instead they are merely code to establish the simulation. These scripts activate the J-DSP applet containing the [Start] button. By pressing this button, the programmed flowgram will appear.

To create J-DSP scripts, an HTML parameter tag, known as the <PARAM> tag is used. The use of this tag is described below. The general format for a <PARAM> tag is:

<PARAM NAME=”*parameter1Name”* VALUE=”*a Value”*>

The first line of the J-DSP script is a <PARAM> tag, which specifies the total number of J-DSP script lines. The parameter name is “numCommand”, and its value is the number of <PARAM>

tags that will be used in this particular applet, excluding itself. This will become clearer as the <PARAM> tag description continues. The format for the first line of the script is:

<PARAM NAME=”numCommand” VALUE=”*number of param tags below”*>

For the general <PARAM> tags that follow, each tag’s PARAM NAME is given a unique number. This number increases sequentially starting from 0. For example,

<PARAM NAME=”0” VALUE=”*a Value”*>
<PARAM NAME=”1” VALUE=”*another Value”*>

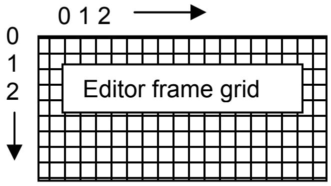
The PARAM NAME, VALUE etc keywords are case insensitive, so it is not necessary to use capital letters.

In the VALUE part of the <PARAM> tag, the parameters **B**, **C**, **O**, **P** and ***** are used to specify different instructions. For example, **B** is used to instruct the editor to create a new part (block). These parameters are further explained in the following sub-sections.

3.2. Creating parts.

One important instruction that can be given to the J-DSP editor is to create a new part, or block as parts are often referred to. Therefore, **B** represents a new block, and is followed by a number *i*, which specifies that this is the *i*th block of the flowgram. This number is also used to refer to the part when performing other tasks with scripts, like opening its dialog box. The block name follows, with the coordinates of the editor frame in parentheses.

For example, **B**0-siggen(3,1) means that the first block of the flowgram is a **SigGen** block and is to be placed in the coordinates (3,1). The editor frame grid is shown below. Each box is approximately 50x70 pixels.



Here is the <PARAM> tag for this case:

<param name=”0” value=”**B**0-siggen(3,1)”>

Now we are ready to write a short and simple program to establish a block in the editor frame:

0	<applet CODE="JDsp.class" width="400" height="250">
1	<param name="numCommand" value="1">
2	<param name="0" value="**B**0-siggen(3,1)">
3	</applet>

Observe that the first <PARAM> tag in line 1 with “numCommand” as PARAM NAME, has VALUE = “1” because there is only one general <PARAM> tag in this program. This is the <PARAM> tag in line 2. Line 2 establishes a ***SigGen*** block at the location given by the coordinates (3,1).

If we include the applet above in an HTML file and open the file with a browser, the J-DSP applet will be activated, and by pressing [Start], the J-DSP editor will load containing a **SigGen** block as shown in figure 4.

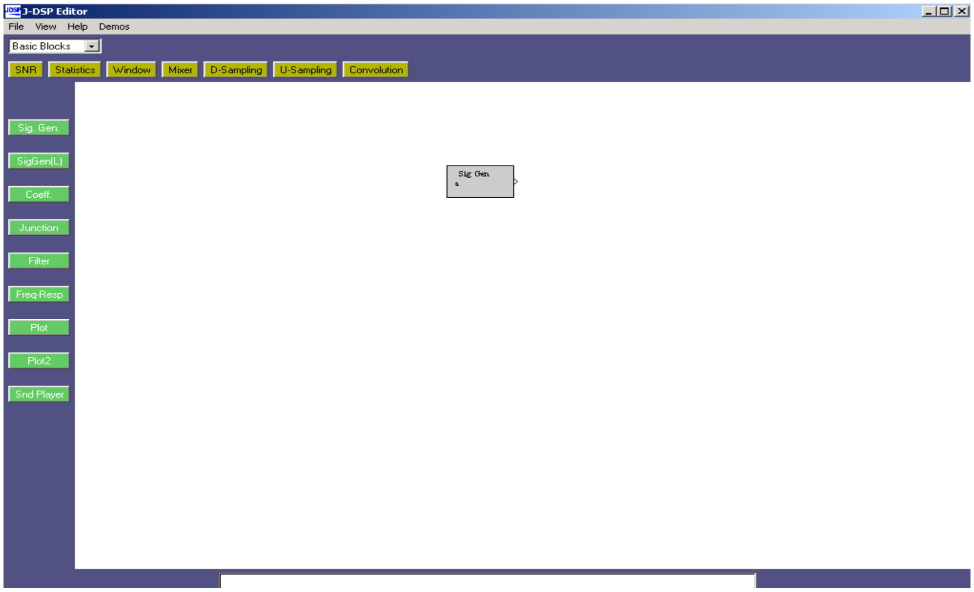
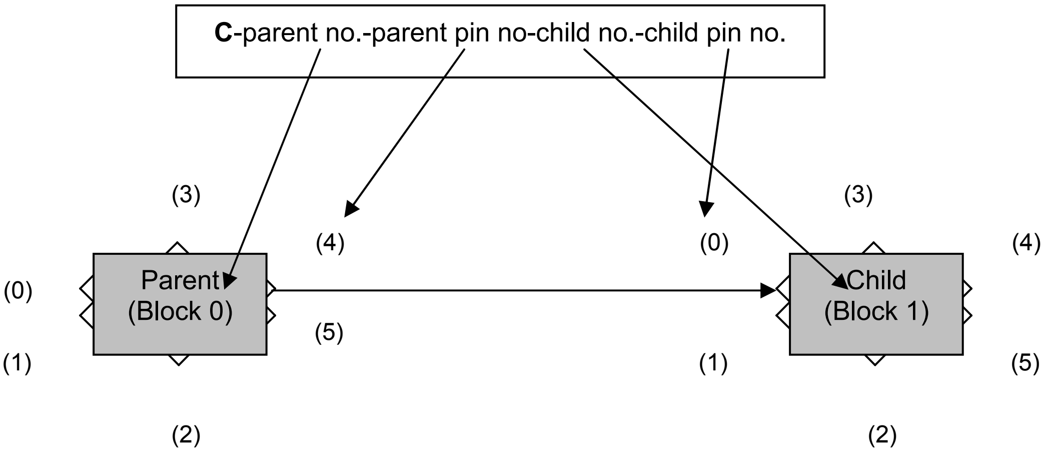


Figure 4: Our first scrip loaded

3.3. Establishing connections.

C is used to denote connections between the blocks. A <PARAM> tag containing **C** in its value will connect two blocks in the same manner as we connect two blocks by dragging from a pin of a block to a pin of another block to connect them. The possible pin numbers for the blocks are shown below.

The format of VALUE in the <PARAM> tag that establishes connection between two blocks has a general format as given below:



The connection shown above requires a code given by C-0-4-1-0. This translates to connecting the first block (number 0) pin 4 to the second block (number 1) pin 0. If we assume this is the 6th general PARAM tag (count starts from 0) the complete line is:

<param name="5" value="**C**-0-4-1-0">

We can now write a simple program to connect two blocks in J-DSP. In the code is given below observe how the <PARAM> tag in line 1 with “numCommand” as PARAM NAME, has VALUE=“3” because there are 3 general <PARAM> tags in this program.

```
0   <applet CODE="JDsp.class" width="400" height="250">

1   <param name="numCommand" value="3">

2   <param name="0" value="B0-siggen(3,1)">
3   <param name="1" value="B1-plot(5,1)">
4   <param name="2" value="C-0-4-1-0">

5   </applet>
```

The above program establishes a **SigGen** block in the (3,1) position and a **Plot** block in the (5,1) position of the editor frame. The last <PARAM> tag connects the two blocks, by connecting **SigGen** block’s 4th pin and **Plot** block’s 0th pin. Opening the html file and starting J-DSP will give the flowgram of figure 5.

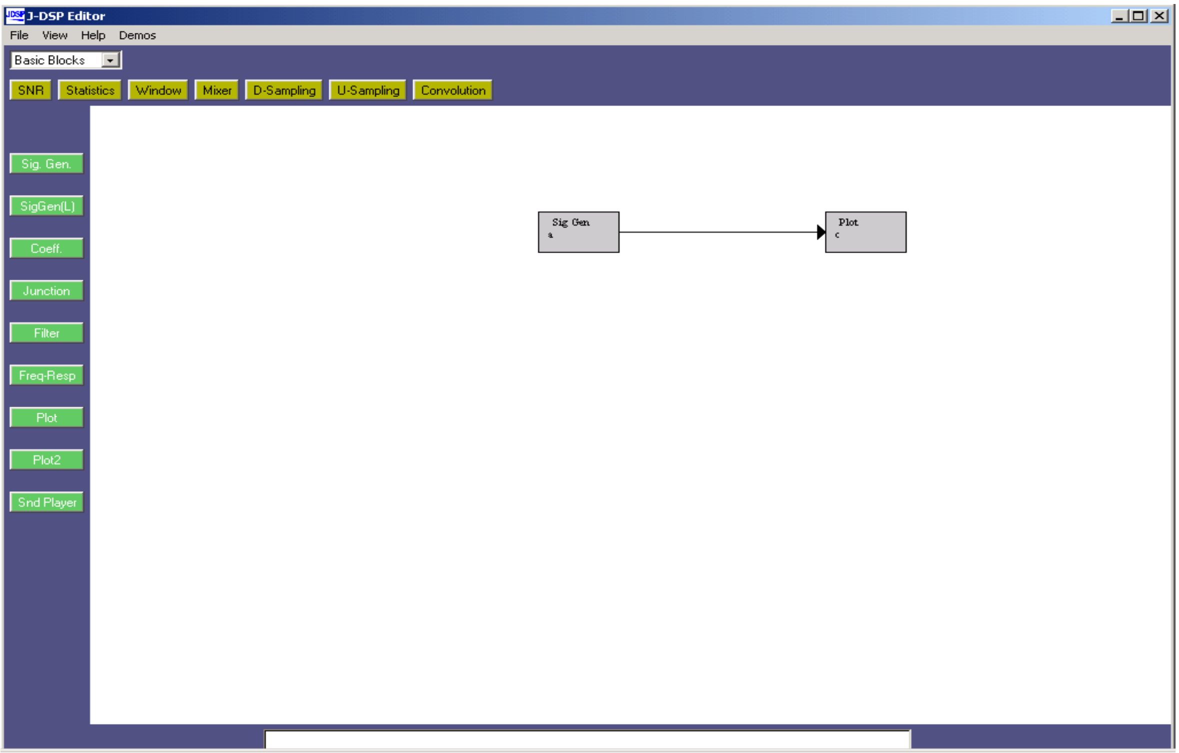


Figure 5: A flowgram with connections

3.4. Passing parameters to parts.

P is used to denote parameter passing in a specific block. A <PARAM> tag containing **P** in its value will cause the set of parameters that follow to be passed to the loaded block. The number concatenated with **P** is the number originally given to the part when created with the **B** command. For example, in the code below, line 3 shows how a signal generator part is created and given the number 0.

```
<param name="0" value="B0-siggen(1,3)">
```

Then, line 9 is used to pass parameters to the signal generator part by placing a 0 next to the P command.

Parameters to be passed to the block

```
<param name="2" value="P0~24,10,0,~3.0,0.9,0.0,0.2,~a,Triangular,Yes,null,~~">
```

Summarizing, **P** denotes that parameters are to be passed, and 0 gives the part number to which they are to be passed to. The set of parameters to be passed to the part then follows.

```
0   <applet CODE="JDsp.class" width="400" height="250">

1   <param name="numCommand" value="3">

2   <!-- START PARTS -->
3   <param name="0" value="B0-siggen(1,3)">
4   <!-- END PARTS -->

5   <!-- START OPEN DIALOGS -->
6   <param name="1" value="O-0">
7   <!-- END OPEN DIALOGS -->

8   <!-- part parameters -->
9   <param name="2" value="P0~24,10,0,~3.0,0.9,0.0,0.2,~a,Triangular,Yes,null,~~">
10  <!-- part parameters -->

11  </applet>
```

The set of parameters passed to each part differs but the form of the parameter passing line is always the same and is given by

<param name="*x*" value="P*y*~$i_0,i_1,\ldots i_n$,~$d_0,d_1,\ldots d_n$,~$s_0,s_1,\ldots s_n$,~$b_0,b_1,\ldots b_n$,~">

where *x* is the script line number and *y* is the number given to the part when it was created. Here, *i* stands for integers (e.g. 2), *d* for doubles (e.g. 3.52), *s* for strings (text) and *b* for Booleans (true or false). Note that this line has to be typed exactly as shown above with the commas (,) and tildes (~) in the exact position even when a certain type of variables is not present (notice that the two consecutive tildes at the end of line 9 are there, even though no Boolean variables are present).

For explanatory purposes table 1 describes the parameters of a signal generator (**SigGen**) block. However, since it could become cumbersome to manually pass parameters to a part, no other tables have been added and the user is urged to automatically generate scripts that pass parameters to parts. This will save both time and debugging efforts. The following paragraphs describe how to use table 1.

As mentioned earlier, the line necessary to pass parameters to a part has the following form:

<param name="*x*" value="P*y*~$i_0,i_1,\ldots i_n$,~$d_0,d_1,\ldots d_n$,~$s_0,s_1,\ldots s_n$,~$b_0,b_1,\ldots b_n$,~">

All the information that is to replace the i, d, s and b is given in table 1 specifically for the signal generator block. The table is divided into four rows, for the variable name, type, position in the code and the allowed range.

The variable name is a short description of the variable, and is not used in the code. The type describes whether the variable is an integer, a double, a string or a Boolean. The position is the exact location where the variable should be placed in the code. It is given by a combination of a letter and a number subscripted after it. The letter can be i, d, s or b for integer, double, string and Boolean respectively. The integer denotes the position, starting with 0. The range is the allowed assortment of values the variable is allowed to take. All variable names with a * and a number in parenthesis depend on a particular selection of signal type but nonetheless should be used.

By replacing the necessary variables in the above generic script line you can get the necessary code to pass parameters to a part.

Variable	Pulse Width	Period	Time shift	Gain (amplitude)	Exponential base (*1)	Mean(*2)	Frequency(*3)	Part name	Signal type	periodic	Distribution(*2)
Type	Integer	Integer	Integer	Double	Double	Double	Double	String	String	String	String
Position	i_0	i_1	i_2	d_0	d_1	d_2	d_3	s_0	s_1	s_2	s_3
Range	1-256	≥ 0	≥ 0	≥ 0	>0	≥ 0.0	≥ 0.0	A 5 digit alphanumeric word	Rectangular Triangular Delta Exponential(*1) Sinusoid(*3) Sinc Random(*2) Self Defined	Yes No	Null Uniform Gaussian Rayleigh

Table 1: Signal Generator parameters table

The parameter passing line can be skipped if no parameters are to be passed to the part, which then loads with its default parameters.

Finally, figure 6 shows a J-DSP script for creating a simulation using a filter and a coefficient block.

```
<applet CODE="JDsp.class" width="400" height="250">

<param name="numCommand" value="12">

<!-- START PARTS -->
<param name="0" value="B0-siggen(1,3)">
<param name="1" value="B1-filter(3,3)">
```

```
<param name="2" value="B2-coeff(3,5)">
<param name="3" value="B3-plot(5,3)">
<!-- END PARTS -->

<!-- START CONNECTIONS -->
<param name="4" value="C-2-3-1-2">
<param name="5" value="C-0-4-1-0">
<param name="6" value="C-1-4-3-0">
<!-- END CONNECTIONS -->

<!-- START OPEN DIALOGS -->
<param name="7" value="O-3">
<!-- END OPEN DIALOGS -->

<!-- part parameters -->
<param name="8" value="P0~20,10,0,~1.0,0.9,0.0,0.2,~a,Rectangular,No,null,~~">
<param name="9" value="P1">
<param name="10" value="P2~~1.0,1.8,-
2.0,3.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,~c,~~">
<param name="11" value="P3~~~d,cont.,Magn.,linear,~false,~">
<!-- part parameters -->

</applet>
```

Figure 6

3.5. Opening Dialog Windows.

O is used to open the dialog box of a block. The effect is the same as double clicking on a block in the J-DSP editor frame. The number of the block has to be specified here. This is the number that we assigned to the block when it was established. For example, **O**-1 opens the dialog box of the second block of the flowgram, because ‘1’ is the number of the second block. The complete line is:

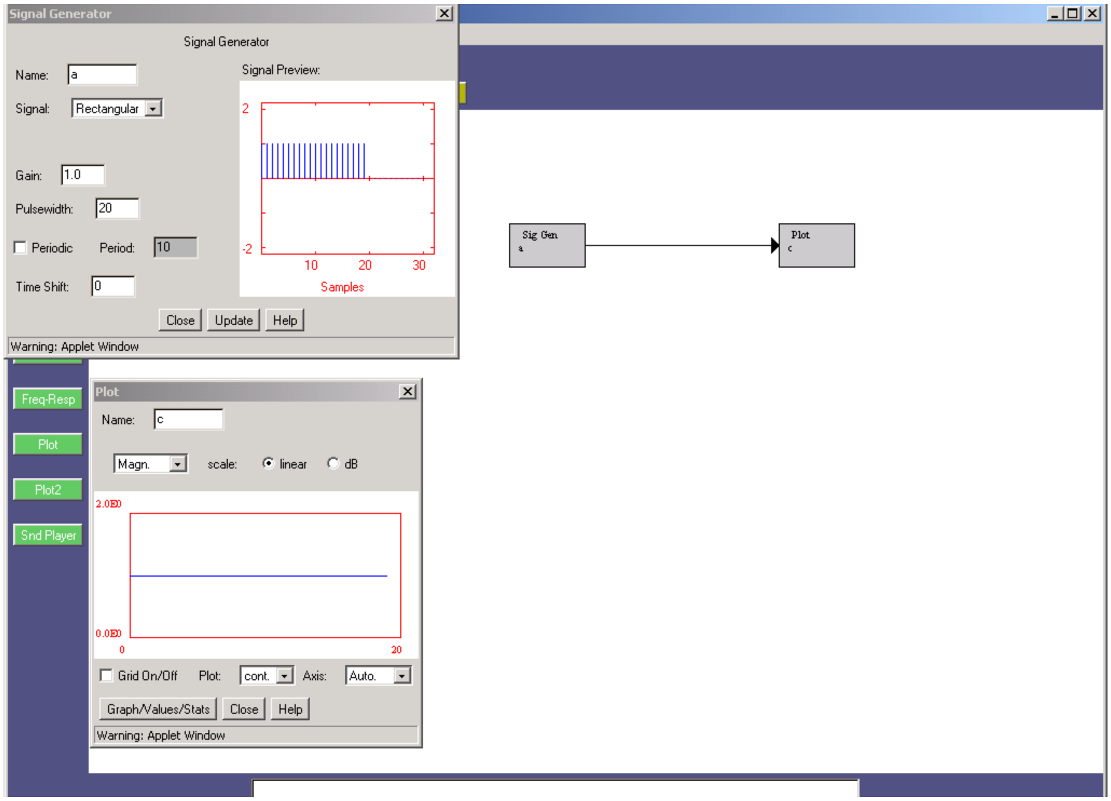
<param name="9" value="**O**-1">

Now we are ready to write a simple program to connect two blocks in J-DSP and open their dialog boxes to observe input and output plots. Here is the necessary code:

0	<applet CODE="JDsp.class" width="400" height="250">
1	<param name="numCommand" value="5">
2	<param name="0" value="**B**0-siggen(3,1)">
3	<param name="1" value="**B**1-plot(5,1)">
4	<param name="2" value="**C**-0-4-1-0">
5	<param name="3" value="**O**-0">
6	<param name="4" value="**O**-1">
7	</applet>

The above program establishes a **Sig Gen** block in the (3,1) position and a **Plot** block in the (5,1) position of the editor frame. The <PARAM> tag in line 4 connects the two blocks, by connecting **Sig Gen** block’s 4th pin and **Plot** block’s 0th pin. The last 2 <PARAM> tags in lines 5 and 6 open the dialog boxes of the **SigGen** block (block number ‘0’) and the **Plot** block (block number ‘1’) respectively. Observe that the <PARAM> tag with “numCommand” as PARAM name, has VALUE= “5” because there are 5 general <PARAM> tags in this program.

Opening the HTML file and starting J-DSP will give the following flowgram, as show in figure 7.


Signal Generator

Name: a

Signal: Rectangular

Gain: 1.0

Pulsewidth: 20

□ Periodic Period: 10

Time Shift: 0

Warning: Applet Window

Plot

Name: c

Magn. scale: ● linear ○ dB

□ Grid On/Off Plot: cont. Axis: Auto.

Warning: Applet Window

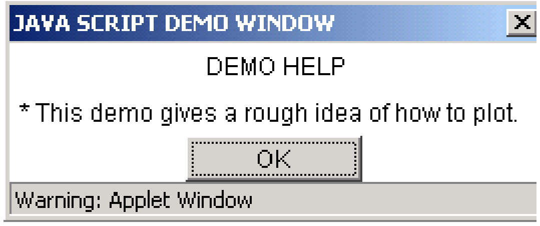
Figure 7: Opening dialog windows

3.6. Help Dialogs.

The asterisk sign (*) followed by any message generates a help dialog box containing the message. An example is given below:

<param name="15" value="* 1. This demo gives a rough idea of how to plot ">

This line establishes the following dialog box in the J-DSP editor:


JAVA SCRIPT DEMO WINDOW

DEMO HELP

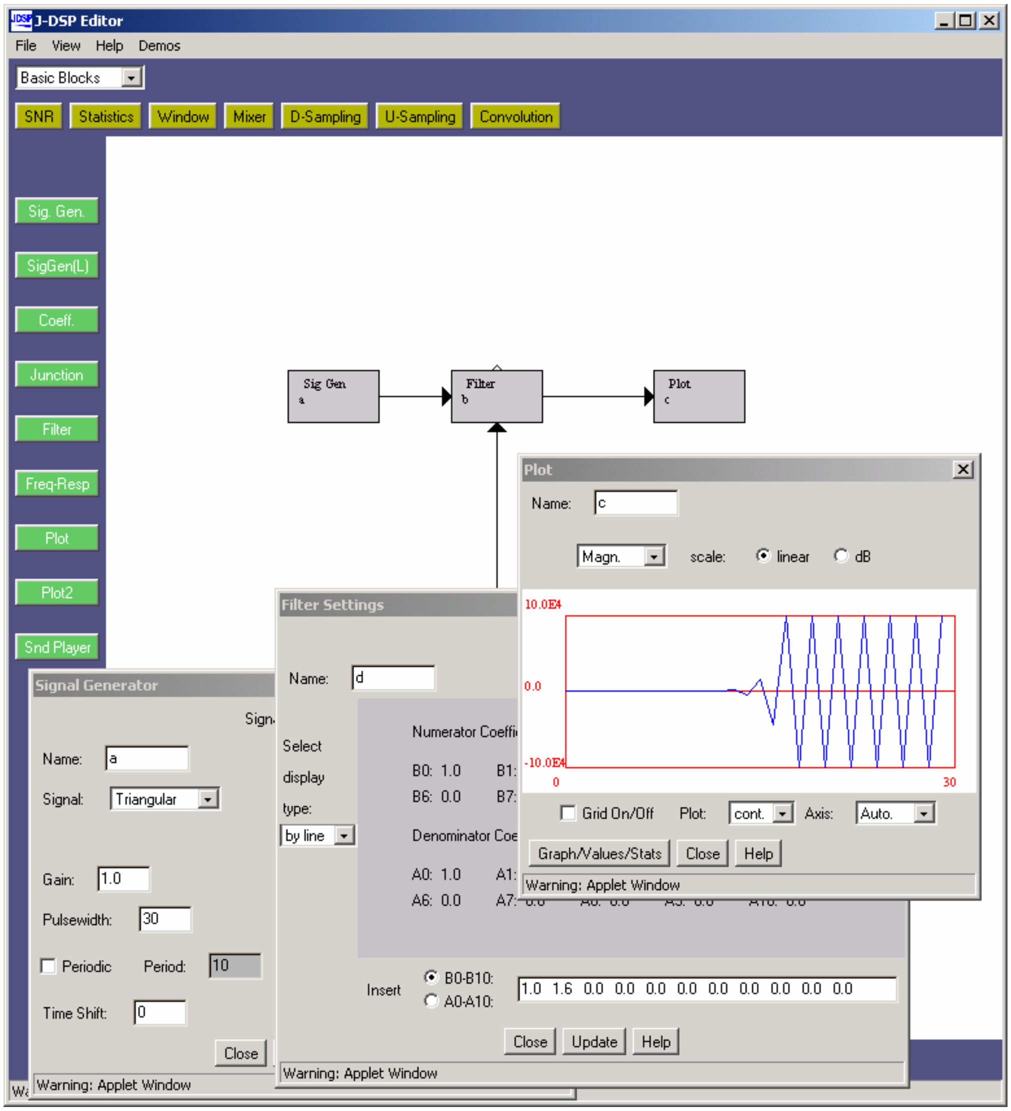
* This demo gives a rough idea of how to plot.

Warning: Applet Window

3.7. An example

The following example includes several of the J-DSP scripts parameters. The resulting J-DSP editor window is shown in figure 8.

```
<applet CODE="JDsp.class" width="400" height="250">

<param name="numCommand" value="14">

<!-- START PARTS -->
<param name="0" value="B0-siggen(1,2)">
<param name="1" value="B1-filter(2,2)">
<param name="2" value="B2-plot(4,2)">
<param name="3" value="B3-coeff(2,5)">
<!-- END PARTS -->

<!-- START CONNECTIONS -->
<param name="4" value="C-0-4-1-0">
<param name="5" value="C-1-4-2-0">
<param name="6" value="C-3-3-1-2">
<!-- END CONNECTIONS -->

<!-- START OPEN DIALOGS -->
<param name="7" value="O-0">
<param name="8" value="O-2">
<param name="9" value="O-3">
<!-- END OPEN DIALOGS -->

<!-- START PART PARAMATERS. * DO NOT MODIFY! * -->
<param name="10" value="P0~30,10,0,~1.0,0.9,0.0,0.2,~a,Triangular,No,null,~~">
<param name="11" value="P1">
<param name="12" value="P2~~~c,linear,Magn.,cont.,~false,~">
<param name="13" value="P3~~1.0,1.6,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,1.0,2.0,-
2.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,~d,~~">
<!-- END PART PARAMATERS -->

</applet>
```



Signal Generator

Name: a

Signal: Triangular

Gain: 1.0

Pulsewidth: 30

☐ Periodic Period: 10

Time Shift: 0

Warning: Applet Window

Filter Settings

Name: d

Select display type: by line

Numerator Coeffi

B0: 1.0 B1:

B6: 0.0 B7:

Denominator Coe

A0: 1.0 A1:

A6: 0.0 A7: 0.0 A8: 0.0 A9: 0.0 A10: 0.0

Insert ● B0-B10: ○ A0-A10: 1.0 1.6 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

Warning: Applet Window

Plot

Name: c

Magn. scale: ● linear ○ dB

☐ Grid On/Off Plot: cont. Axis: Auto.

Warning: Applet Window

Figure 8: The result of the example script

Section M9   Audio effects blocks

These blocks appear at the top of the simulation area

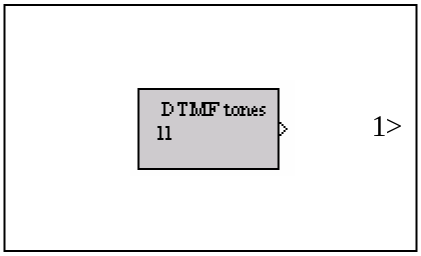
Table of blocks	
Block notation	Description
DTMF tones	Generates Dual Tone Multi Frequency tones
MIDI	Generates MIDI sounds
Echo	Generates echo effect of the input signal
Reverb	Generates reverb effect of the input signal
Reverb Demo	Simulates five specific reverb types

M9.1

Block name : DTMF tones **Notation**: ***DTMF tones***

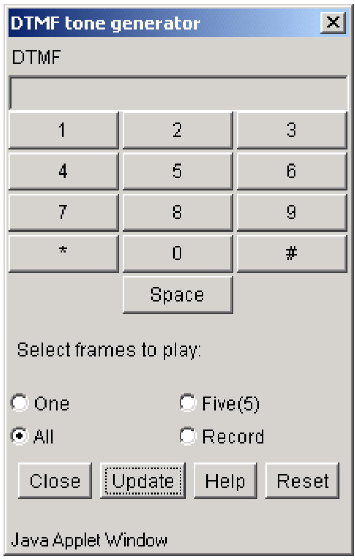
Description: Generates dual-tone-multi-frequency (DTMF) tones used in landline telephony applications. This block generates a single tone of length: 256 (1 frame), 1280 (5 frames) and 8192 (32 frames) samples. It also generates a sequence of pre-recorded tones. The sampling frequency is 8KHz. The tones can be played back using the J-DSP provided sound player, and used in a DSP simulation.

Pin assignment:



Pin	Description
1	DTMF tone
2	
3	
4	
5	
6	

Dialog window(s):



Select frames to play:

○ One ○ Five(5)

● All ○ Record

(a)DTMF tones dialog window

Script use:

Name: DTMF

Example code: <param name="3" value="B0-DTMF(3,1)">

Equation(s) Implemented :

$$y = \cos(2p\, f_1 nT) + \cos(2p\, f_2 nT)$$

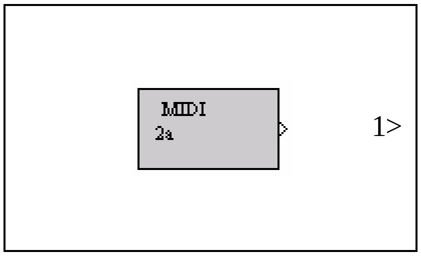
where f_1 and f_2 are chosen from the tone frequencies (697, 770, 852, 941, 1209, 1336, 1477 (Hz)). The sampling frequency is 8 KHz, i.e., T = 0.125ms

M9.2

Block name : MIDI **Notation**: ***MIDI***

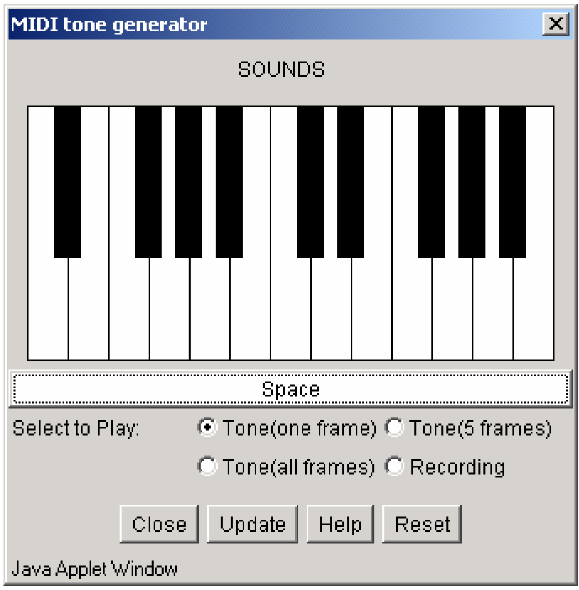
Description: Simulates a piano keyboard and generates Musical Instrument Digital Interface (MIDI) sounds at the frequencies described by the MIDI standard. The MIDI block can generate a single tone of length: 256 (1 frame), 1280 (5 frames) and 8192 (32 frames) samples. It can also generate a sequence of pre-recorded tones. The sampling frequency is 8KHz. All the tones can be used in a DSP simulation and are audible using the J-DSP provided sound player.

Pin assignment:



Pin	Description
1	MIDI tone
2	
3	
4	
5	
6	

Dialog window(s):



Select to Play: ● Tone(one frame) ○ Tone(5 frames) ○ Tone(all frames) ○ Recording

(a)MIDI dialog window

Script use:

Name: MIDI

Example code: <param name="1" value="B1-MIDI(2,1)">

Equation(s) Implemented :

$$y = \cos(2p\ fnT)$$

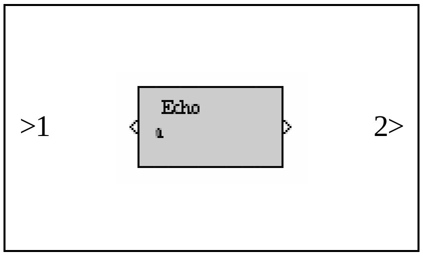
where *f* is taken from a MIDI standard table [www.midi.org]

M9.3

Block name : Echo **Notation**: ***Echo***

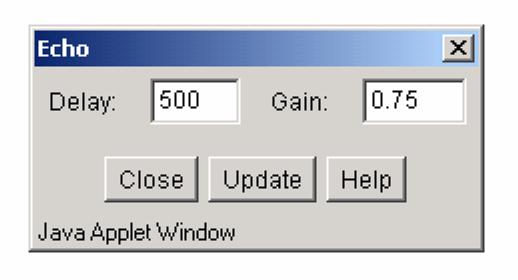
Description: This block generates the echo effect of the input signal. The echo effect is obtained by mixing the input signal with its delayed version. The proportion of the delayed signal to the "clean" original signal determines how obvious the echo is, and the delay signifies the echo period.

Pin assignment:



Pin	Description
1	Input time-domain signal
2	Output signal with echo
3	
4	
5	
6	

Dialog window(s):



Delay: 500 Gain: 0.75

(a)Echo dialog window

Script use:
Name: Echo
Example code: <param name="2" value="B2-Echo(1,4)">

Equation(s) Implemented :

$$y(n)= x(n) + b\, x(n-R)$$

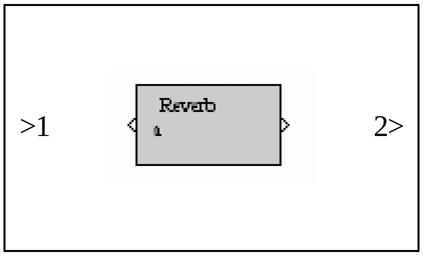
R = the number of echo delay in samples. In order to have a distinguishable echo, R should be relatively large. b is the attenuation constant ($|b| < 1$). Recomended values to perceive an echo are b=0.75 and R=500

M9.4

Block name : Reverberation **Notation**: ***Reverb***

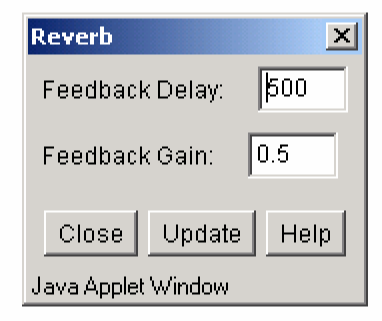
Description: This block implements a reverberation effect on the input signal. Reverberation is obtained by mixing the input signal with the delayed versions of its feedback. The effect of the feedback results in multiple echos.

Pin assignment:



Pin	Description
1	Input time-domain signal
2	Output signal
3	
4	
5	
6	

Dialog window(s):



Feedback Delay: 500

Feedback Gain: 0.5

(a)Reverb dialog window

Script use:

Name: Reverb

Example code: <param name="3" value="B3-Reverb(2,4)">

Equation(s) Implemented :

$$y(n) = x(n) + b\ \ y(n-R)$$

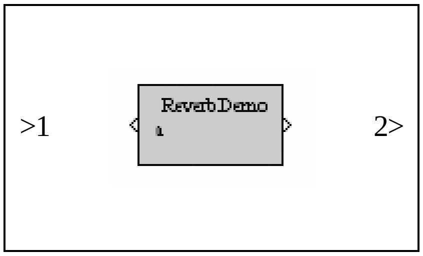
R = feedback delay in samples. b is the attenuation constant ($|b| < 1$).

M9.5

Block name : Reverberation Demo **Notation**: ***Reverb.Demo***

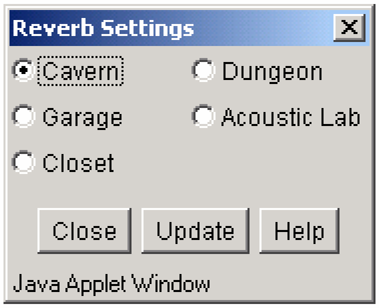
Description: This block is a demonstration of the reverberation effect, simulating five specific cases given by, “Cavern” (delay=600, gain=0.7), “Dungeon” (delay=160, gain=0.8), “Garage” (delay=240, gain=0.4), “Acoustic Lab” (delay=128, gain=0.6) and “Closet” (delay=40, gain=0.1).

Pin assignment:



Pin	Description
1	Input time-domain signal
2	Output signal
3	
4	
5	
6	

Dialog window(s):



● Cavern ○ Dungeon
○ Garage ○ Acoustic Lab
○ Closet

(a)Reverb.Demo dialog window

Script use:

Name: ReverbDemo
Example code: <param name="4" value="B4-ReverbDemo(4,3)">

Equation(s) Implemented :

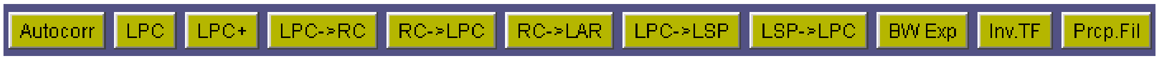
$$y(n) = x(n) + b.\, y(n-R)$$

R = feedback delay in samples. b is the attenuation constant ($|b| < 1$).

Section M8: Speech blocks

These blocks appear at the top of the simulation area

Table of blocks	
Block notation	Description
Autocorr	Computes the autocorrelation sequence of the input signal
LPC	Calculates the linear predictor coefficients (LPC)
LPC+	Computes the LP coefficients
LPC -> RC	Converts the LP coefficients to reflection coefficients (RC)
RC -> LPC	Converts reflection coefficients to LP coefficients
RC-> LAR	Computes the log-area-ratio values (LARs)
LPC ->LSP	Converts LP coefficients to line spectral pairs (LSP)
LSP->LPC	Computes LP coefficients from the LSP
BW Exp	Function to expand the bandwidth of the filter
Inv.TF	Reciprocates the input transfer function
Prcp.Fil	Performs perceptual weighted filtering



M8.1

Block name : Autocorrelation **Notation**: ***Autocorr***

Please refer to section M7, block M7.1

M8.2

Block name : LP coefficients **Notation**: ***LPC***

Please refer to section M7, block M7.2

M8.3

Block name : LP coefficients + **Notation**: ***LPC+***

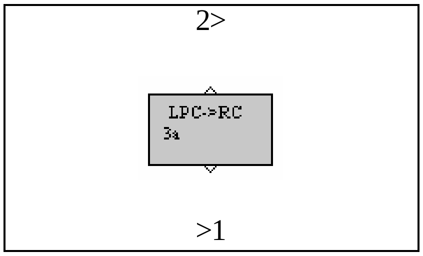
Please refer to section M7, block M7.3

M8.4

Block name : LPC to RC **Notation**: ***LPC->RC***

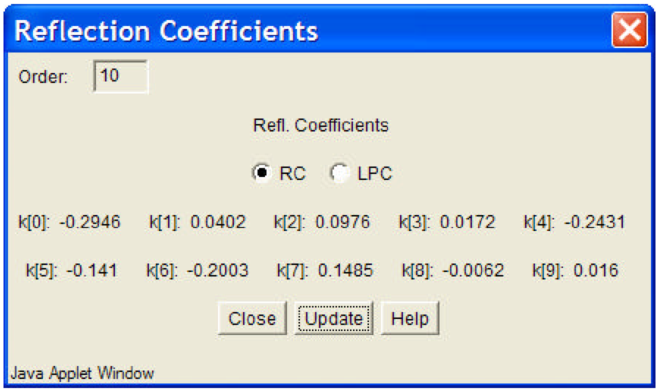
Description: This block converts the direct-form LP coefficients (a_i) to reflection coefficients (k_i). The Levinson recursion algorithm is used to implement the LPC to RC conversion. A check-box option is provided to view the LP coefficients and reflection coefficients.

Pin assignment:



Pin	Description
1	LP coefficients of order 10, a_i
2	Reflection coefficients, k_i
3	
4	
5	
6	

Dialog window(s):



Order: 10

Refl. Coefficients

● RC ○ LPC

k[0]: -0.2946 k[1]: 0.0402 k[2]: 0.0976 k[3]: 0.0172 k[4]: -0.2431

k[5]: -0.141 k[6]: -0.2003 k[7]: 0.1485 k[8]: -0.0062 k[9]: 0.016

(a)LPC->RC dialog window

Script use:

Name: lpc2rc

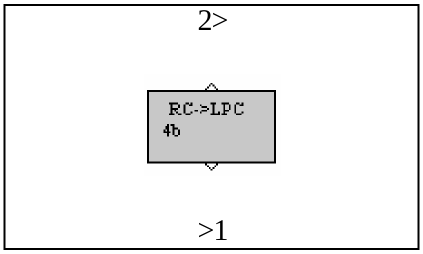
Example code: <param name = “3” value = “B3-lpc2rc(3,1)”>

M8.5

Block name : RC to LPC **Notation**: ***RC->LPC***

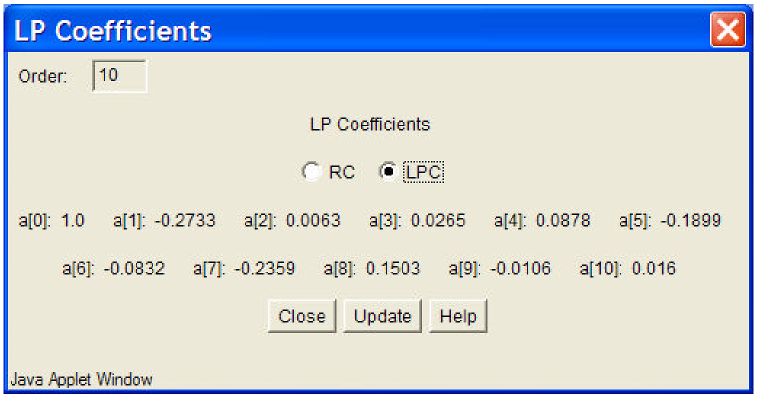
Description: This block computes the LP coefficients (a_i) from the reflection coefficients (k_i).

Pin assignment:



Pin	Description
1	Reflection coefficients, k_i
2	LP coefficients of order 10, a_i
3	
4	
5	
6	

Dialog window(s):



LP Coefficients

Order: 10

LP Coefficients

○ RC ● LPC

a[0]: 1.0 a[1]: -0.2733 a[2]: 0.0063 a[3]: 0.0265 a[4]: 0.0878 a[5]: -0.1899

a[6]: -0.0832 a[7]: -0.2359 a[8]: 0.1503 a[9]: -0.0106 a[10]: 0.016

(a)RC->LPC dialog window

Script use:

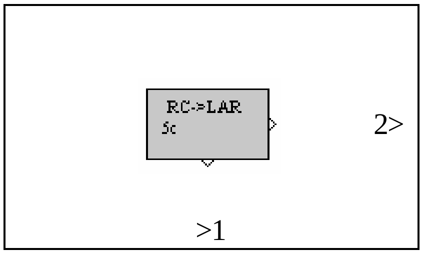
Name: rc2lpc
Example code: <param name = “3” value = “B3-rc2lpc(3,1)”>

M8.6

Block name : RC to LAR **Notation**: ***RC->LAR***

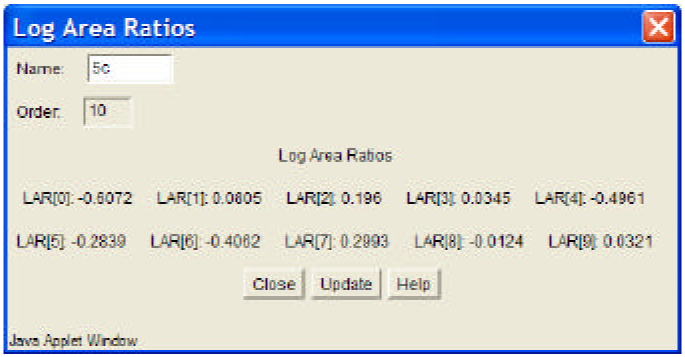
Description: This block converts the reflection coefficients to log area ratios (LARs).

Pin assignment:



Pin	Description
1	Reflection coefficients, k_i
2	Log area ratios (LARs)
3	
4	
5	
6	

Dialog window(s):



Log Area Ratios

Name: 5c

Order: 10

Log Area Ratios

LAR[0]: -0.6072 LAR[1]: 0.0805 LAR[2]: 0.196 LAR[3]: 0.0345 LAR[4]: -0.4961

LAR[5]: -0.2839 LAR[6]: -0.4062 LAR[7]: 0.2993 LAR[8]: -0.0124 LAR[9]: 0.0321

(a)RC->LAR dialog window

Script use:

Name: rc2lar

Example code: <param name = “3” value = “B3-rc2lar(3,1)”>

Equation(s) Implemented :

$$LAR(i) = \left(\frac{1+k_i}{1-k_i} \right)$$

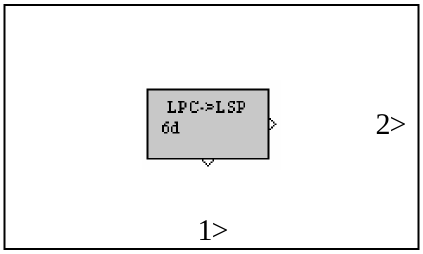
where $k_{i\,=}$ reflection coefficients, *LAR(i)* = Log area ratio (i)

M8.7

Block name: LPC to LSP **Notation**: ***LPC->LSP***

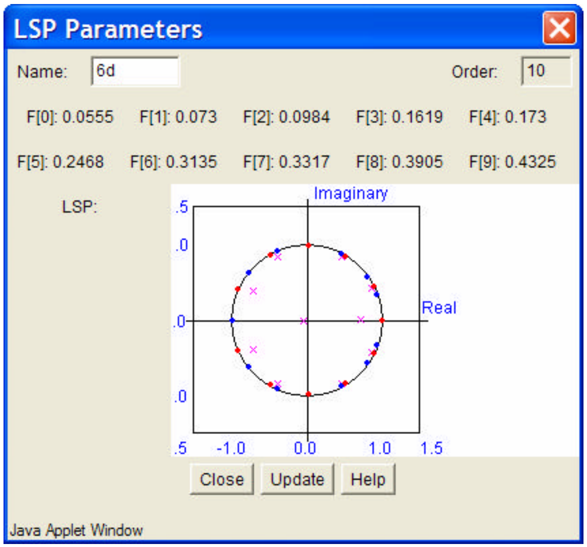
Description: This block computes the line spectral pairs (LSP) from the LP coefficients.

Pin assignment:



Pin	Description
1	LP coefficients, a_i
2	Line spectral pairs, F_i
3	
4	
5	
6	

Dialog window(s):


LSP Parameters

Name: 6d Order: 10

F[0]: 0.0555 F[1]: 0.073 F[2]: 0.0984 F[3]: 0.1619 F[4]: 0.173

F[5]: 0.2468 F[6]: 0.3135 F[7]: 0.3317 F[8]: 0.3905 F[9]: 0.4325

(a)LPC->LSP dialog window

Script use:

Name: lpc2lsp

Example code: <param name = “3” value = “B3-lpc2lsp(3,1)”>

Equation(s) Implemented :

The sum polynomial F_1 (z) is given by, $F_1(z) = \dfrac{A(z) + z^{-11} A(z^{-1})}{1 + z^{-1}}$

The difference polynomial F_2 (z) is given by, $F_2(z) = \dfrac{A(z) - z^{-11} A(z^{-1})}{1 - z^{-1}}$

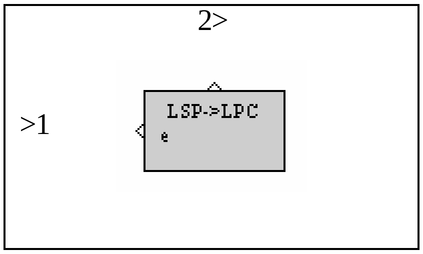
Each polynomial has five conjugate roots on the unit circle and they alternate each other.

M8.9

Block name : LSP to LPC **Notation**: ***LSP->LPC***

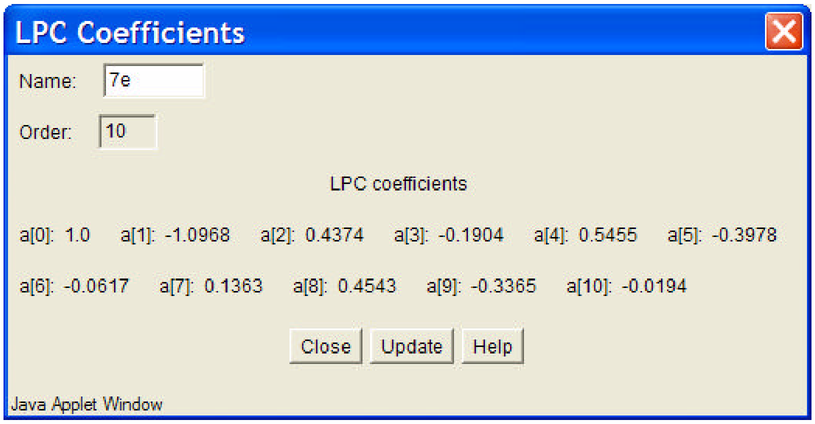
Description: This block computes the LP coefficients from the line spectral pairs.

Pin assignment:



Pin	Description
1	Line spectral pairs, F_i
2	LP coefficients, a_i
3	
4	
5	
6	

Dialog window(s):



Name: 7e

Order: 10

LPC coefficients

a[0]: 1.0 a[1]: -1.0968 a[2]: 0.4374 a[3]: -0.1904 a[4]: 0.5455 a[5]: -0.3978

a[6]: -0.0617 a[7]: 0.1363 a[8]: 0.4543 a[9]: -0.3365 a[10]: -0.0194

(a)LSP->LPC dialog window

Script use:

Name: lsp2lpc
Example code: <param name = "3" value = "B3-lsp2lpc(3,1)">

Equation(s) Implemented :

$$A(z) = \frac{F_1(z) + F_2(z)}{2}$$

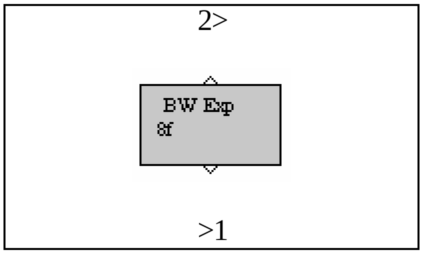
where, $F_1(z)$ = sum polynomial, $F_2(z)$ = difference polynomial, and $A(z)$ = LP filter

M8.10

Block name : Bandwidth expansion **Notation**: ***BW. Exp.***

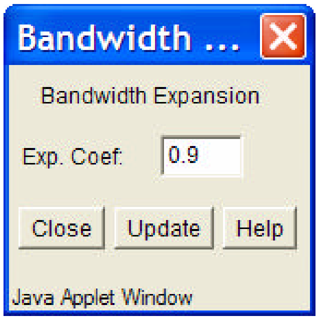
Description: This block performs the bandwidth expansion operation.

Pin assignment:



Pin	Description
1	Filter coefficients
2	Bandwidth expanded filter coefficients
3	
4	
5	
6	

Dialog window(s):



Bandwidth Expansion

Exp. Coef: 0.9

Java Applet Window

(a)BW. Exp. dialog window

Script use:

Name: BWExp
Example code: <param name = "3" value = "B3-BWExp(3,1)">

Equation(s) Implemented :

Input filter transfer function, $$H(z) = \frac{b_0 + b_1 z^{-1} + b_2 z^{-2} ... + b_{10} z^{-10}}{1 + a_1 z^{-1} + a_2 z^{-2} ... + a_{10} z^{-10}}$$

Bandwidth expanded filter, $$H_B(z) = \frac{b_0 + b_1 g z^{-1} + b_2 g^{2} z^{-2} ... + b_{10} g^{10} z^{-10}}{1 + a_1 g z^{-1} + a_2 g^{2} z^{-2} ... + a_{10} g^{10} z^{-10}}$$

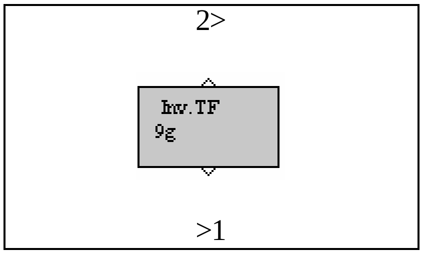
where g is the bandwidth expansion coefficient.

M8.11

Block name : Inverse Transfer Function **Notation**: ***Inv. TF***

Description: This block inverts the transfer function at its input.

Pin assignment:



Pin	Description
1	Filter coefficients
2	Inverse transformed transfer function
3	
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: Inv.TF
Example code: <param name = “3” value = “B3-Inv.TF(3,1)”>

Equation(s) Implemented :

Input filter, $H(z) = \dfrac{b_0 + b_1 z^{-1} + b_2 z^{-2} ... + b_{10} z^{-10}}{1 + a_1 z^{-1} + a_2 z^{-2} ... + a_{10} z^{-10}}$

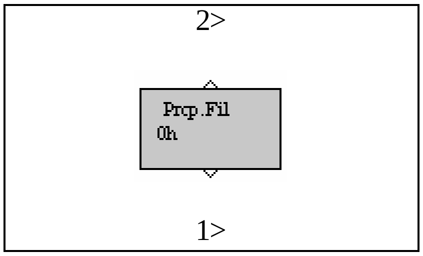
Inverse transformed transfer function, $H_I(z) = \dfrac{(1 + a_1 z^{-1} + a_2 z^{-2} ... + a_{10} z^{-10}) / b_0}{1 + (b_1 z^{-1} + b_2 z^{-2} ... + b_{10} z^{-10}) / b_0}$

M8.12

Block name : Perceptual weighted filtering **Notation**: ***Prcp.Fil.***

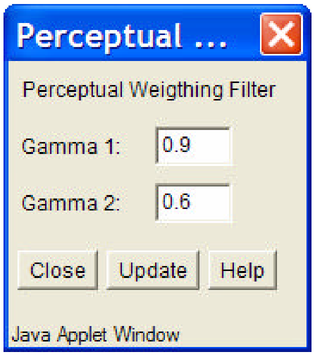
Description: This block performs the perceptual weighted filtering or simply perceptual weighting. The weights ?$_1$, ?$_2$ can be entered by the user.

Pin assignment:



Pin	Description
1	LP coefficients, A(z)
2	Perceptual weighted output, *W(z)*
3	
4	
5	
6	

Dialog window(s):


Perceptual ...

Perceptual Weigthing Filter

Gamma 1: 0.9

Gamma 2: 0.6

Java Applet Window

(a)Prcp.Fil dialog window

Script use:

Name: Prcp.Fil

Example code: <param name = “3” value = “B3-Prcp.Fil(3,1)”>

Equation(s) Implemented :

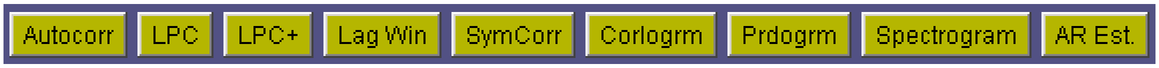
Perceptual weighting filter is given by $W(z) = \dfrac{A(z/g_1)}{A(z/g_2)} = \dfrac{1+\sum_{i=1}^{10} g_1{}^i a_i z^{-i}}{1+\sum_{i=1}^{10} g_2{}^i a_i z^{-i}}$

g_1, g_2 are the perceptual weights, and a_i are the LP coefficients.

Section M7: Statistical DSP blocks

These blocks appear at the top of the simulation area

Table of blocks	
Block notation	Description
Autocorr	Computes the autocorrelation values of a signal
LPC	Computes the linear predictor coefficients (LPC)
LPC+	Computes the linear predictor coefficients (LPC)
Lag Win	Windows a time-domain signal
Sym Corr	Finds the symmetric autocorrelation
Corlogrm	PSD estimation using Correlogram method
Prdogrm	PSD estimation using Periodogram method
Spectrogram	Provides frequency versus time plots
AR Est.	AR estimation based on the Levinson-Durbin algorithm

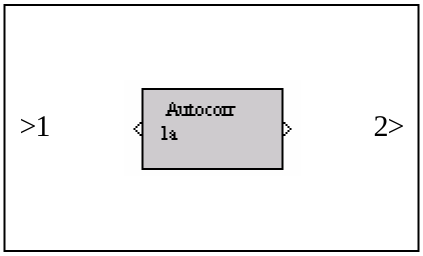


M7.1

Block name : Autocorrelation **Notation**: ***Autocorr***

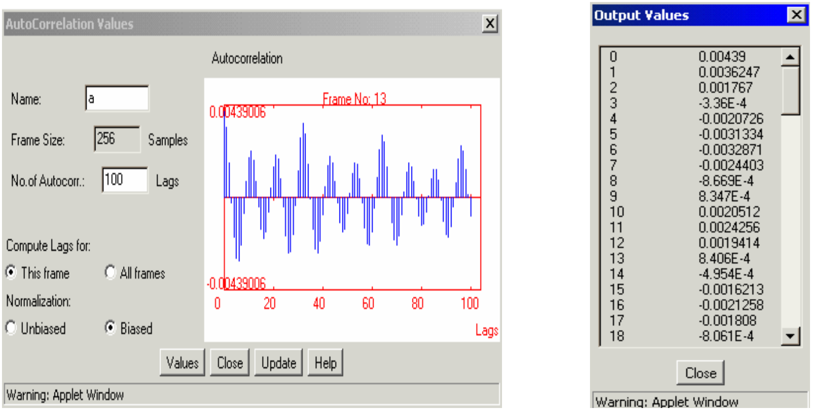
Description: This block calculates the autocorrelation sequence of a signal. The user needs to specify the number of lags and select whether they are computed for a particular frame ("this frame") or for "all frames". An option for "biased" or "unbiased" normalization is provided.

Pin assignment:



Pin	Description
1	Time-domain signal *x(n)*
2	Autocorrelation $r_{xx}(m)$
3	
4	
5	
6	

Dialog window(s):



Name: a

Frame Size: 256 Samples

No.of Autocorr.: 100 Lags

Compute Lags for:

● This frame ○ All frames

Normalization:

○ Unbiased ● Biased

```
0     0.00439
1     0.0036247
2     0.001767
3     -3.36E-4
4     -0.0020726
5     -0.0031334
6     -0.0032871
7     -0.0024403
8     -8.669E-4
9     8.347E-4
10    0.0020512
11    0.0024256
12    0.0019414
13    8.406E-4
14    -4.954E-4
15    -0.0016213
16    -0.0021258
17    -0.001808
18    -8.061E-4
```

(a)Autocorrelation dialog window and output values

Script use:

Name: autocorr
Example code: <param name = "3" value = "B3-autocorr(3,1)">

Equation(s) Implemented :

$$r_{xx}(m) = \frac{1}{L}\sum_{n=0}^{N-m-1} x^*(n+m)x(n)$$

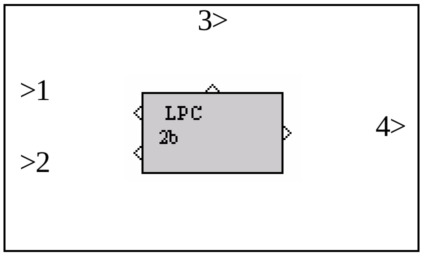
where, *m* is the number of lags; *0 = m = N-1*
If *L* = *N*, a biased autocorrelation sequence is obtained
If *L* = *N* – *m*, an unbiased autocorrelation sequence is obtained

M7.2

Block name: Linear prediction coefficients **Notation**: ***LPC***

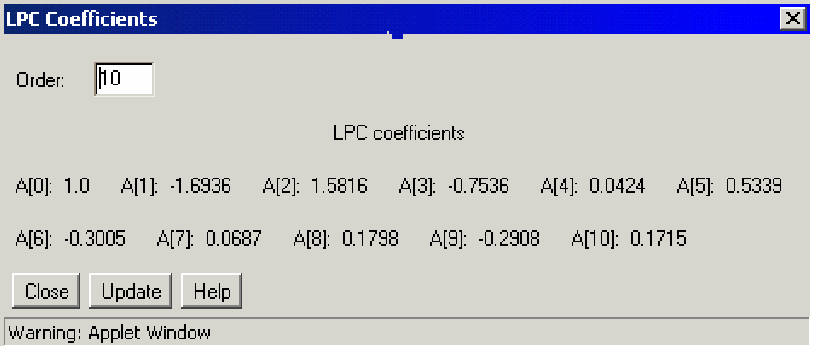
Description: This block computes the linear predictor coefficients (LPC) based on the Levinson-Durbin algorithm.

Pin assignment:



Pin	Description
1	Time-domain signal, *x(n)*
2	Autocorrelation sequence, $r_{xx}(m)$
3	LP coefficients, a_i
4	Residual signal, *e(n)*
5	
6	

Dialog window(s):



Order: 10

LPC coefficients

A[0]: 1.0 A[1]: -1.6936 A[2]: 1.5816 A[3]: -0.7536 A[4]: 0.0424 A[5]: 0.5339

A[6]: -0.3005 A[7]: 0.0687 A[8]: 0.1798 A[9]: -0.2908 A[10]: 0.1715

(a)LPC dialog window

Script use:

Name: LPC
Example code: <param name = “3” value = “B3-LPC(3,1)”>

Equation(s) Implemented :

$$\text{Residual signal is given by, } e(n) = x(n) - \sum_{i=1}^{p} a_i x(n-i)$$

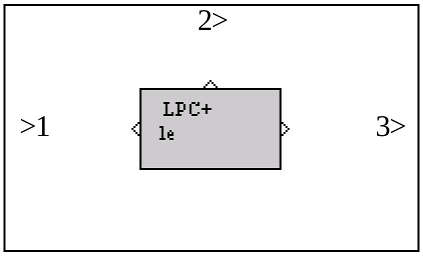
$$\text{LP synthesis filter is given by, } H(z) = \frac{1}{1 + \sum_{i=1}^{p} a_i z_i^{-i}}$$

M7.3

Block name : Linear prediction coefficients + **Notation**: ***LPC+***

Description: This block calculates the linear predictor coefficients (LPC). The autocorrelation function is incorporated in this block in contrast to the LPC block.

Pin assignment:



Pin	Description
1	Time-domain signal, *x(n)*
2	LP coefficients, a_i
3	Residual signal, *e(n)*
4	
5	
6	

Dialog window(s):

- None-

Script use:

Name: LPC+
Example code: <param name = “3” value = “B3-LPC+(3,1)”>

Equation(s) Implemented :

Residual signal is obtained by using the equation, $e(n) = x(n) - \sum_{i=1}^{p} a_i x(n-i)$

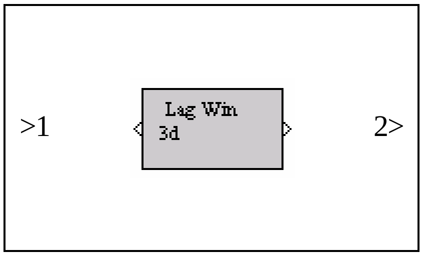
LP synthesis filter is given by, $H(z) = \dfrac{1}{1 + \sum_{i=1}^{p} a_i z_i^{-i}}$

M7.4

Block name : Lag window **Notation**: ***Lag. Win***

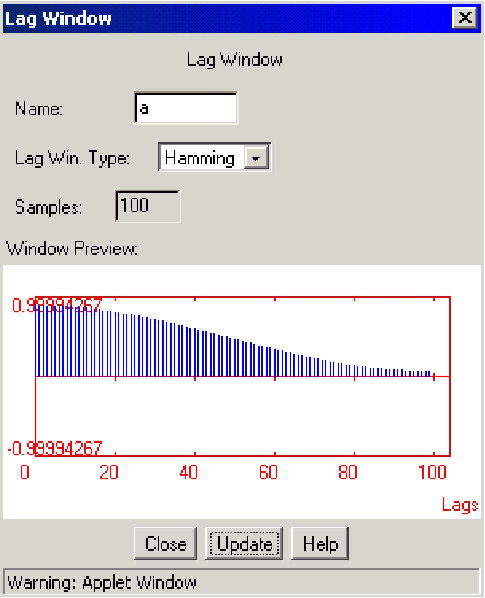
Description: This block windows the input signal with a user-defined window function. The window functions available are: Hamming, Hanning, rectangular, Bartlett, Blackman, and Kaiser. The maximum window length is 256 samples.

Pin assignment:



Pin	Description
1	Autocorrelation sequence, $r_{xx}(m)$
2	Windowed autocorrelation, $r^{w}(m)$
3	
4	
5	
6	

Dialog window(s):



Lag Window

Name: a

Lag Win. Type: Hamming

Samples: 100

Window Preview:

Warning: Applet Window

(a)Lag. Win dialog window

Script use:

Name: lagwindow

Example code: <param name = “3” value = “B3-lagwindow(3,1)”>

Equation(s) Implemented :

$$r^{w}_{xx}(m) = w(m)r_{xx}(m)$$

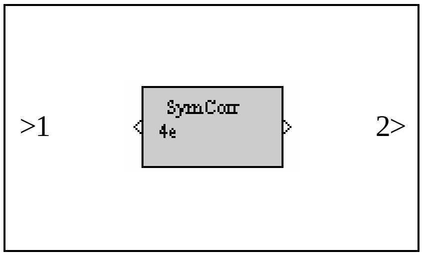
$r_{xx}(m)$ is the autocorrelation sequence and $r^{w}_{xx}(m)$ the windowed autocorrelation.

M7.5

Block name : Symmetric correlation **Notation**: ***Sym. Corr.***

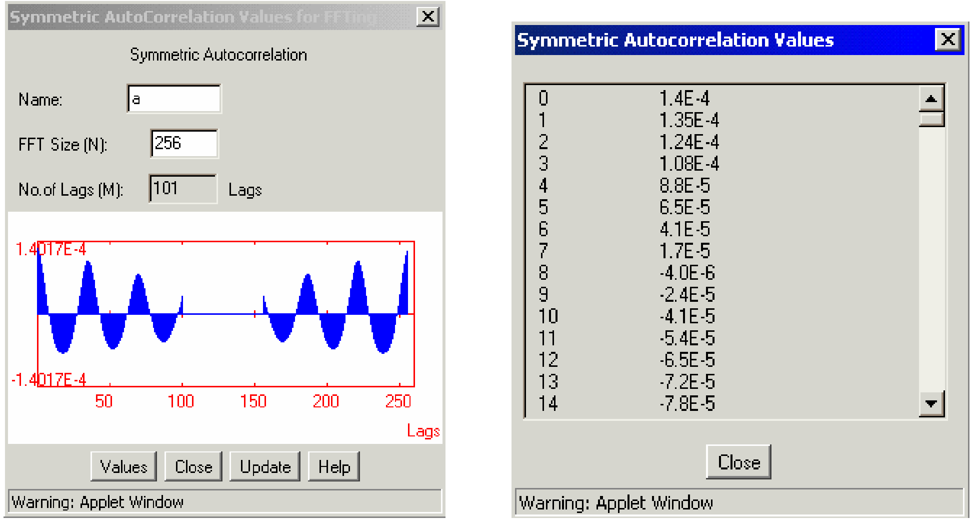
Description: This block makes the autocorrelation lags, r_{xx} symmetric so that they can be used with the FFT block in order to calculate the power spectral density (PSD). Symmetry of the autocorrelation sequence around 0 is modified to symmetry around the edges

Pin assignment:



Pin	Description
1	Autocorrelation sequence, $r_{xx}(m)$
2	Symmetric autocorrelation sequence, $r_{xx}^{(s)}(m)$
3	
4	
5	
6	

Dialog window(s):


Symmetric AutoCorrelation Values for FFTing
1.4017E-4
-1.4017E-4
50
100
150
200
250
Lags
Values
Close
Update
Help
Symmetric Autocorrelation Values
Close

Symmetric Autocorrelation

Name: a

FFT Size (N): 256

No.of Lags (M): 101 Lags

Warning: Applet Window

```
0     1.4E-4
1     1.35E-4
2     1.24E-4
3     1.08E-4
4     8.8E-5
5     6.5E-5
6     4.1E-5
7     1.7E-5
8     -4.0E-6
9     -2.4E-5
10    -4.1E-5
11    -5.4E-5
12    -6.5E-5
13    -7.2E-5
14    -7.8E-5
```

Warning: Applet Window

(a)Sym.Corr. dialog window and output values

Script use:

Name: symcorr
Example code: <param name = “3” value = “B3-symcorr(3,1)”>

Equation(s) Implemented:

$$r^{(s)}_{xx}(N-m) = r_{xx}(m);$$

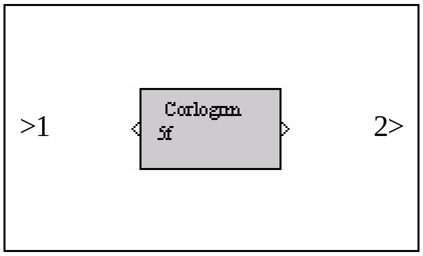
where n = FFT size and m = number of lags
For example if the FFT size, N = 8, and the number of lags is 3, then
$r^{(s)}_{xx}(8) = r_{xx}(0)$, $r^{(s)}_{xx}(7) = r_{xx}(1)$, $r^{(s)}_{xx}(6) = r_{xx}(2)$, and so on.

M7.6

Block name : Correlogram **Notation**: ***Correlogram***

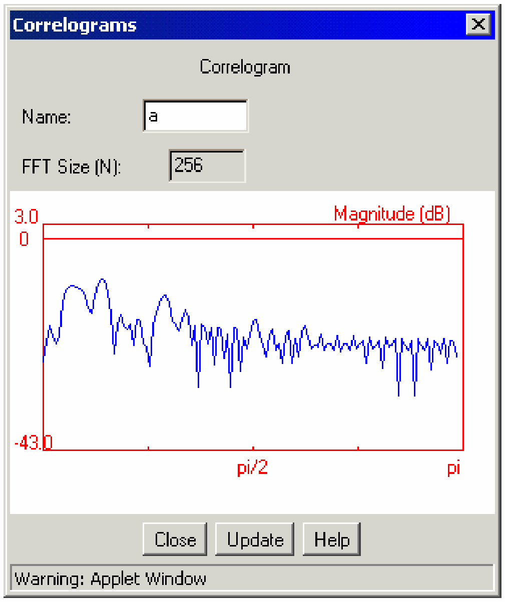
Description: This block computes a PSD estimate by performing an FFT on the symmetric autocorrelation sequence.

Pin assignment:



Pin	Description
1	Symmetric autocorrelation sequence, $r_{xx}^{(s)}(m)$
2	PSD estimate, $R_{xx}(k)$
3	
4	
5	
6	

Dialog window(s):



Name: a

FFT Size (N): 256

Warning: Applet Window

(a)Correlogram dialog window

Script use:

Name: corrlog

Example code: <param name = “3” value = “B3-corrlog(3,1)”>

Equation(s) Implemented :

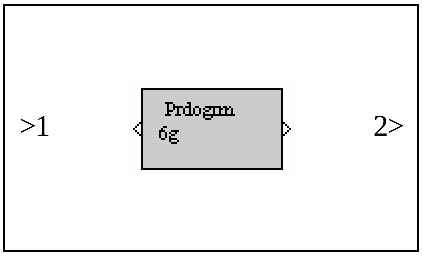
$$R_{xx}(k) = \frac{1}{N}\left[\sum_{m=0}^{N-1} \mathrm{r}^{(s)}{}_{xx}\,(\mathrm{m})e^{-\left[\frac{j2p\,km}{\mathrm{N}}\right]}\right]$$

N = the length of the sequence

Block name : Periodogram **Notation**: ***Prdogm*.**

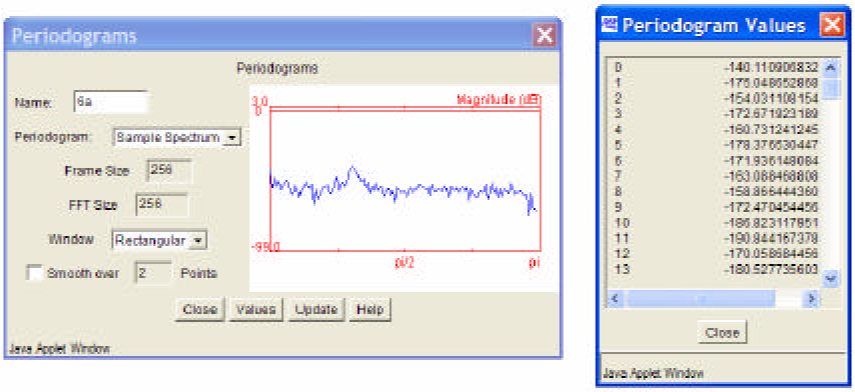
Description: This block estimates the power spectral density (PSD) by operating directly on the data set. Two different periodograms can be used to estimate the PSD: sample spectrum or Welch periodogram. The user can specify the number of "smooth over" points to implement the Daniell periodogram over the sample or the Welch periodograms.

Pin assignment:



Pin	Description
1	Input *x(n)*
2	PSD estimate, $R_{xx}(k)$
3	
4	
5	
6	

Dialog window(s):



Name: 6a
Periodogram: Sample Spectrum
Frame Size 256
FFT Size 256
Window Rectangular
□ Smooth over 2 Points

0	-140.110906832
1	-175.048652868
2	-154.031108154
3	-172.671923189
4	-180.731241245
5	-178.376530447
6	-171.936148084
7	-163.088468808
8	-158.866444360
9	-172.470454456
10	-186.823117851
11	-190.844167378
12	-170.058684456
13	-180.527735603

(a) Prdogm dialog window and output values

Script use:

Name: periodgm

Example code: <param name = "3" value = "B3-periodgm(3,1)">

Equation(s) Implemented :

The sample spectrum of the p^{th} frame is given by, $R_{xx}^{p}(k) = \frac{1}{N}\left|\sum_{n=0}^{N-1} w(n)x^{p}(n)e^{-\frac{j2pkn}{N}}\right|^{2}$,

Welch periodogram, $R_{xx}^{w}(k) = \frac{1}{P}\sum_{p=1}^{P} R_{xx}^{p}(k)$

w(n) = window, $x^{p}(n)$ = The p^{th} frame of the time-domain input signal.

$R_{xx}^{w}(k)$ = Welch PSD estimate of all the frames.

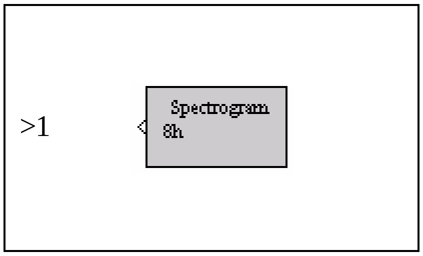
$R_{xx}^{p}(k)$ = Sample PSD estimate of the p^{th} frame.

M7.8

Block name : Spectrogram **Notation**: ***Spectrogram***

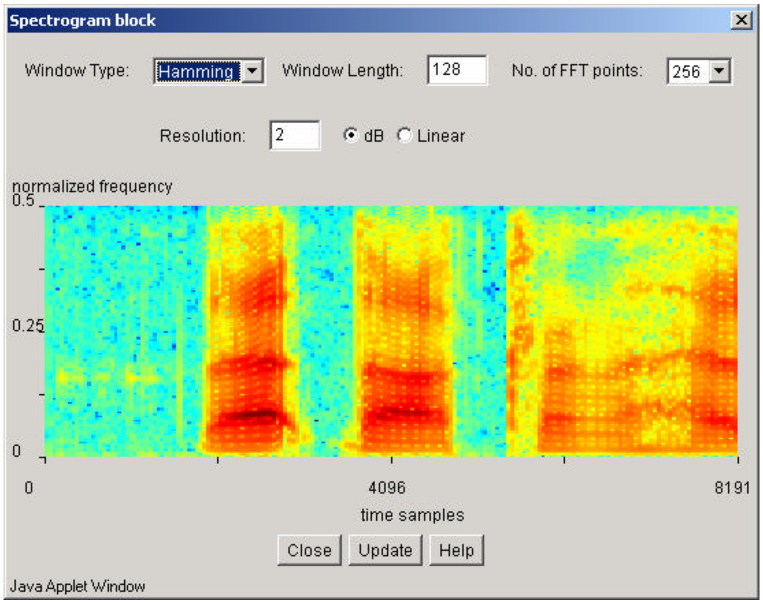
Description: This block calculates the spectrogram (frequency versus time plot) of the given input signal. The window types available are: Hamming, Hanning, rectangular, Gaussian, Bartlett, and Kaiser. The window length, the number of FFT points and the resolution can be specified by the user. By moving the cursor on the plot, the normalized magnitude, and the x-y coordinates can be viewed.

Pin assignment:



Pin	Description
1	Time-domain signal, *x(n)*
2	
3	
4	
5	
6	

Dialog window(s):



Spectrogram block

Window Type: Hamming Window Length: 128 No. of FFT points: 256

Resolution: 2 ● dB ○ Linear

(a) Spectrogram dialog window

Script use:

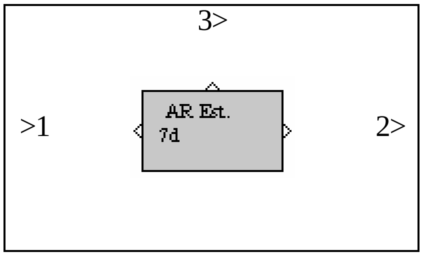
Name: specgram
Example code: <param name="3" value="B2-specgram(3,1)">

M7.10

Block name : AR estimator **Notation**: ***AR Est.***

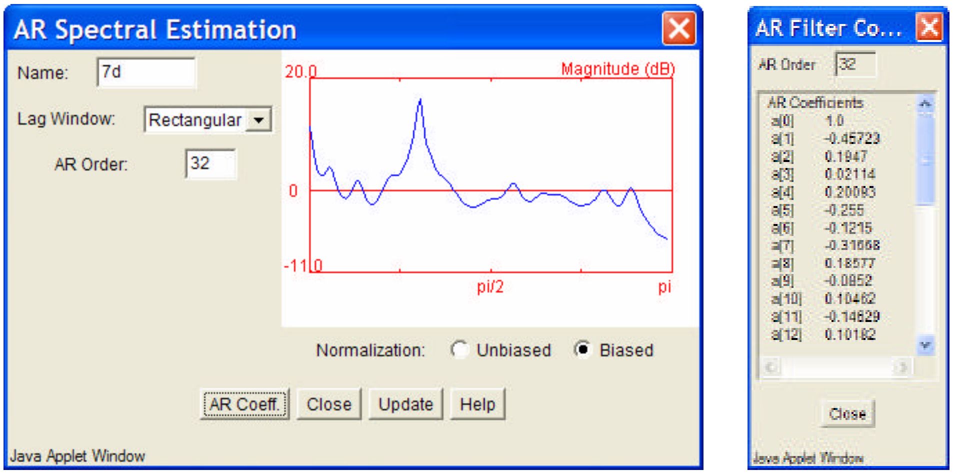
Description: This block computes the AR coefficients and plots the auto-regressive spectrum of the input signal using the Levinson-Durbin algorithm. The following lag windows are available: rectangular, Hamming, triangular, and Gaussian. The maximum number of AR coefficients allowed = 64.

Pin assignment:



Pin	Description
1	Time-domain signal, *x(n)*
2	LPC spectrum, $R^{AR}_{xx}(k)$
3	AR coefficients, a_i
4	
5	
6	

Dialog window(s):



Name: 7d

Lag Window: Rectangular

AR Order: 32

Normalization: ○ Unbiased ● Biased

AR Order 32

AR Coefficients

a[0]	1.0
a[1]	-0.45723
a[2]	0.1947
a[3]	0.02114
a[4]	0.20093
a[5]	-0.255
a[6]	-0.1215
a[7]	-0.31668
a[8]	0.18577
a[9]	-0.0852
a[10]	0.10462
a[11]	-0.14629
a[12]	0.10182

(a) AR Est. dialog window

Script use:

Name: AREst

Example code: <param name = "3" value = "B3-AREst(3,1)">

Equation(s) Implemented :

$$R_{xx}^{AR}(k) = \left| \frac{1}{1 + \sum_{i=1}^{P} a_i z^{-i}} \right|^2$$

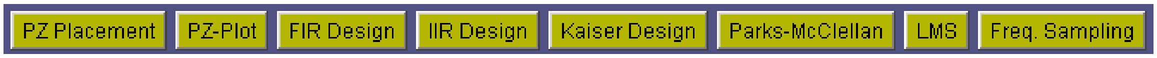
Here, a_i = Linear Prediction (LP) coefficients and N is the order of the LP filter

Section M6: Filter blocks

These blocks appear at the top of the simulation area

Table of blocks

Block notation	Description
PZ-Placement	Allows entering pole/zero values
PZ-Plot	Plots poles/zeros in polar coordinates
FIR Design	FIR filter design
IIR Design	IIR filter design
Kaiser	Kaiser filter design
Parks-McClellan	Parks-McClellan filter design
LMS	LMS adaptive filter algorithm
Freq Samp.	Frequency sampling

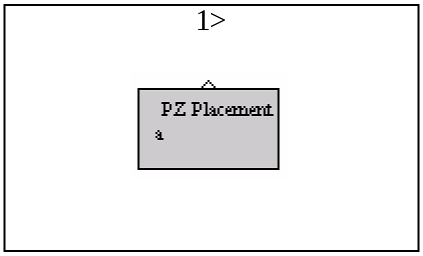


M6.1

Block name : Pole Zero Placement **Notation**: ***PZ-Placement***

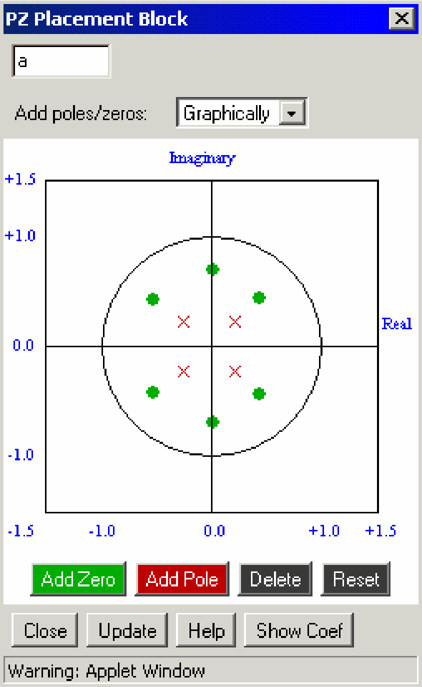
Description: This block allows the user to enter poles and zeros representing a filter. The corresponding filter coefficients are passed to the output. Poles and zeros are added as conjugate pairs, and no more than 10 (5 pairs) can be entered. They can be placed either graphically or manually. Graphical manipulation of poles and zeros is achieved through buttons that allow placing, moving and deleting. Manually placing poles and zeros can be done either in square or polar form.

Pin assignment:



Pin	Description
1	Filter coefficients
2	
3	
4	
5	
6	

Dialog window(s):


PZ Placement Block

a

Add poles/zeros: Graphically

Warning: Applet Window

(a)PZ-Placement dialog window

Script use:

Name: pzplace

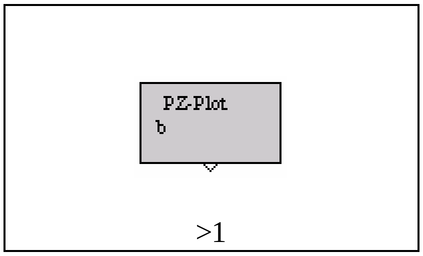
Example code: <param name = “3” value = “B3-pzplace(3,1)”>

M6.2

Block name : Pole-Zero plot **Notation**: ***PZ-Plot***

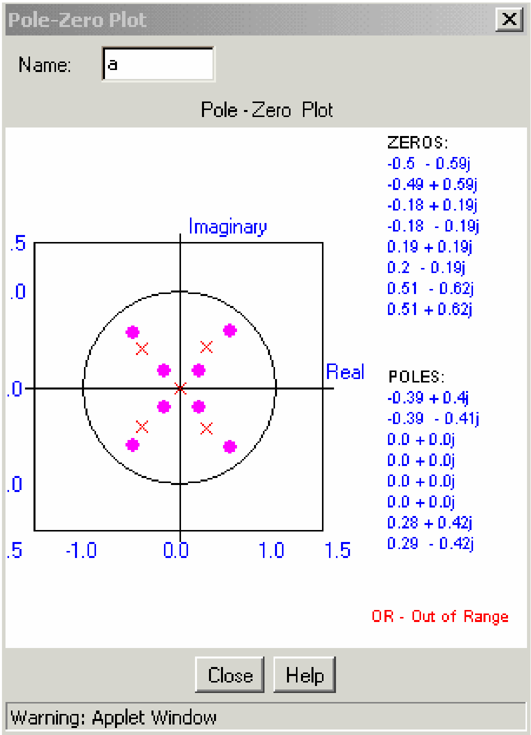
Description: This block calculates and displays the poles and zeros of a transfer function in the z-plane. The block accepts filter coefficients at its input.

Pin assignment:



Pin	Description
1	Filter coefficients
2	
3	
4	
5	
6	

Dialog window(s):



Name: a

```
ZEROS:
-0.5  - 0.59j
-0.49 + 0.59j
-0.18 + 0.19j
-0.18  - 0.19j
0.19 + 0.19j
0.2  - 0.19j
0.51  - 0.62j
0.51 + 0.62j

POLES:
-0.39 + 0.4j
-0.39  - 0.41j
0.0 + 0.0j
0.0 + 0.0j
0.0 + 0.0j
0.0 + 0.0j
0.28 + 0.42j
0.29  - 0.42j

OR - Out of Range
```

Warning: Applet Window

(a)PZ-Plot dialog window

Script use:

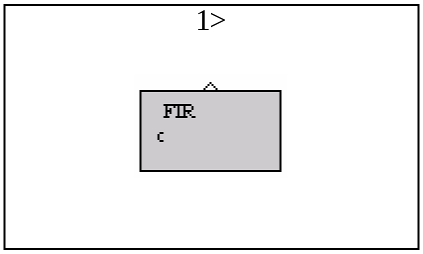
Name: pzplot
Example code: <param name = “3” value = “B3-pzplot(3,1)”>

M6.3

Block name : FIR design **Notation**: ***FIR***

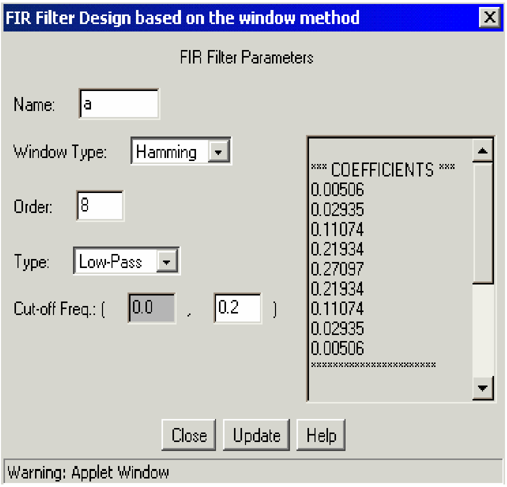
Description: Designs a finite impulse response (FIR) filter based on the windowing method. The windowing FIR filter design method is a straightforward technique implemented by expanding the frequency response of an ideal filter in a Fourier series and then truncating and smoothing the response using a window. The user needs to supply the following information: *Window type*: Hamming, Hanning, Blackman, Bartlett, rectangular or Kaiser | *Filter order* (maximum is 64) | *Type*: low-pass, high-pass, pass-band, or stop-band. Cut-off frequencies (f_c), take values from 0 to 1, where $f_c = 1$ corresponds to half-the-sampling frequency.

Pin assignment:



Pin	Description
1	Filter coefficients
2	
3	
4	
5	
6	

Dialog window(s):


FIR Filter Design based on the window method

FIR Filter Parameters

Name: a

Window Type: Hamming

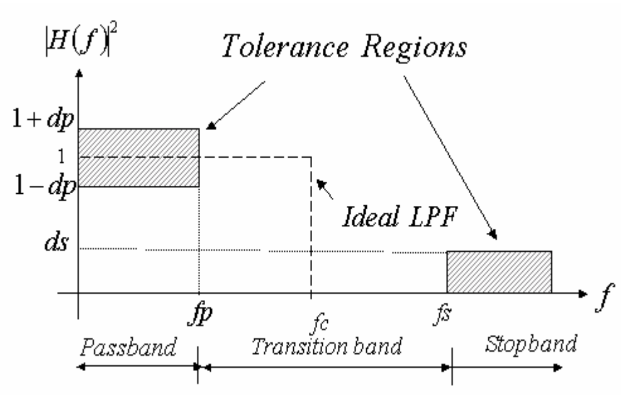
Order: 8

Type: Low-Pass

Cut-off Freq.: (0.0 , 0.2)

```
*** COEFFICIENTS ***
0.00506
0.02935
0.11074
0.21934
0.27097
0.21934
0.11074
0.02935
0.00506
************************
```

Warning: Applet Window



(a)FIR dialog window and filter design specifications

Script use:

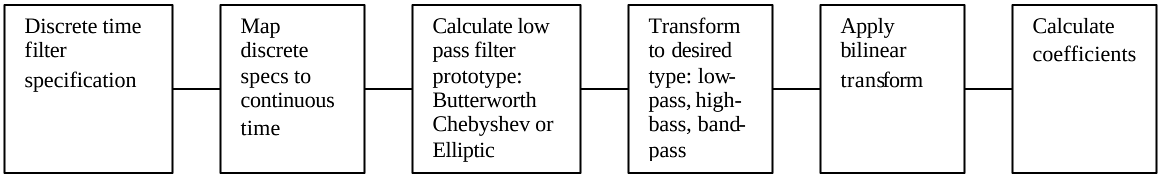
Name: FIR

Example code: <param name = “3” value = “B3-FIR(3,1)”>

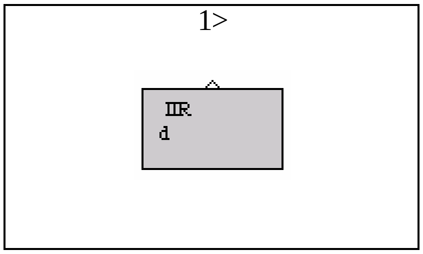
M6.4

Block name : IIR design **Notation**: ***IIR***

Description: Designs an infinite (length) impulse response (IIR) filter based on the bilinear transformation. Butterworth, Chebyshev -I & -II, and Elliptic filters are supported. The filter specifications are in terms of: *Filter typ*e- can be low-pass, high-pass or pass-band | Wp_1, Ws_1 – pass-band and stop-band edge cut-off frequencies respectively, | Wp_2, Ws_2 – second pass-band and stop-band edge cut-off frequencies respectively (for pass-band filters) | *PB*, *SB* – pass-band and stop-band tolerances in dB. Cut-off frequencies (f_c), take values from 0 to 1, where $f_c = 1$ corresponds to half-the-sampling frequency. The design process is illustrated in terms of the block diagram below:

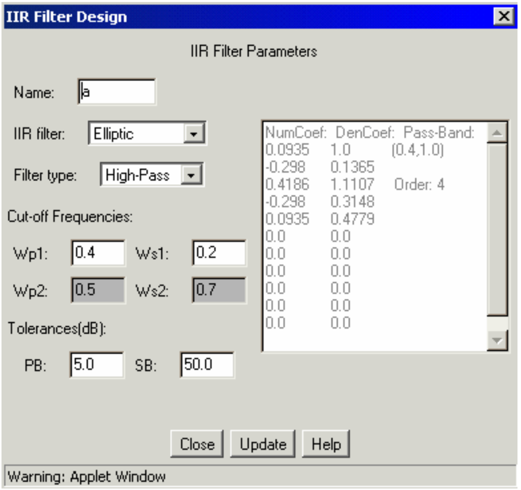

Discrete time filter specification
Map discrete specs to continuous time
Calculate low pass filter prototype: Butterworth Chebyshev or Elliptic
Transform to desired type: low-pass, high-bass, band-pass
Apply bilinear transform
Calculate coefficients

Pin assignment:



Pin	Description
1	Filter coefficients
2	
3	
4	
5	
6	

Dialog window(s):



Name: a

IIR filter: Elliptic

Filter type: High-Pass

Cut-off Frequencies:

Wp1: 0.4 Ws1: 0.2

Wp2: 0.5 Ws2: 0.7

Tolerances(dB):

PB: 5.0 SB: 50.0

```
NumCoef: DenCoef:  Pass-Band:
0.0935   1.0       (0.4,1.0)
-0.298   0.1365
0.4186   1.1107    Order: 4
-0.298   0.3148
0.0935   0.4779
0.0      0.0
0.0      0.0
0.0      0.0
0.0      0.0
0.0      0.0
0.0      0.0
```

(a)FIR dialog window

Script use:

Name: IIR

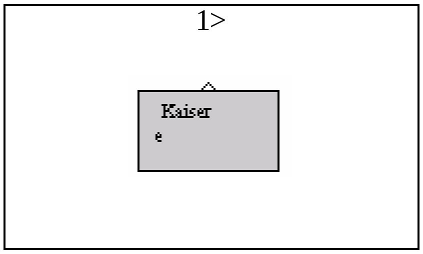
Example code: <param name = “3” value = “B3-IIR(3,1)”>

M6.5

Block name : Kaiser design **Notation**: ***Kaiser***

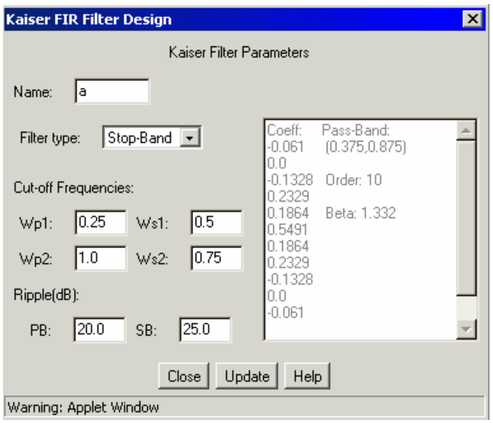
Description: This block designs Kaiser FIR filters based on the windowing method. The design process involves calculating the Fourier series of the ideal filter and then multiplying it with a Kaiser window that best fits the filter specifications. Filter specifications are: *Filter type*: can be low-pass, high-pass, stop-band or pass-band | Wp_1, Ws_1 – pass-band and stop-band edge cut-off frequencies respectively, | Wp_2, Ws_2 – second pass-band and stop-band edge cut-off frequencies respectively (for pass-band filters) | *PB*, *SB* – pass-band and stop-band tolerances in dB.

Pin assignment:



Pin	Description
1	Filter coefficients
2	
3	
4	
5	
6	

Dialog window(s):



Kaiser Filter Parameters

Name: a

Filter type: Stop-Band

Cut-off Frequencies:

Wp1: 0.25 Ws1: 0.5

Wp2: 1.0 Ws2: 0.75

Ripple(dB):

PB: 20.0 SB: 25.0

```
Coeff:   Pass-Band:
-0.061    (0.375,0.875)
0.0
-0.1328  Order: 10
0.2329
0.1864   Beta: 1.332
0.5491
0.1864
0.2329
-0.1328
0.0
-0.061
```

Warning: Applet Window

(a)Kaiser dialog window

Script use:

Name: Kaiser

Example code: <param name = “3” value = “B3-Kaiser(3,1)”>

Equation(s) Implemented :

The order and value of b of the Kaiser window are calculated by:

$$N = \frac{A-8}{2.285\Delta w} \quad \text{and} \quad b = \begin{cases} 0.1102(A-8.7)..A > 50 \\ 0.5842(A-21)^{0.4} + 0.07886(A-21)....21 \le A \le 50 \\ 0...A < 21 \end{cases}$$

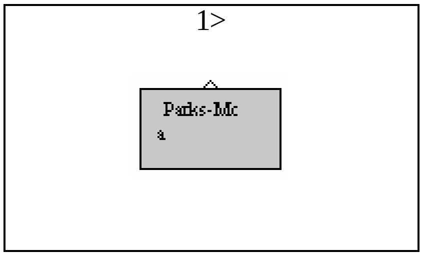
$\Delta\omega$ is the transition band of the filter and A is equal to the smaller of PB and SB.

M6.6

Block name : Parks-McClellan **Notation**: ***Parks-Mc***

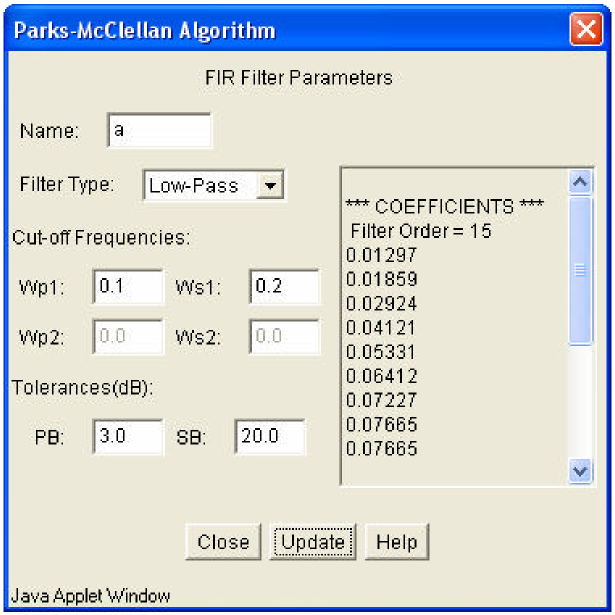
Description: This block designs FIR filters using the Parks-McClellan algorithm with min-max design. Filter specifications are: *Filter type*: can be low-pass, high-pass, stop-band, or pass-band | Wp_1, Ws_1 – pass-band and stop-band edge cut-off frequencies respectively, | Wp_2, Ws_2 – second pass-band and stop-band edge cut-off frequencies respectively (for pass-band filters) | *PB*, *SB* – pass-band and stop-band tolerances in dB.

Pin assignment:



Pin	Description
1	Filter coefficients
2	
3	
4	
5	
6	

Dialog window(s):


Parks-McClellan Algorithm

FIR Filter Parameters

Name: a

Filter Type: Low-Pass

Cut-off Frequencies:

Wp1: 0.1 Ws1: 0.2

Wp2: 0.0 Ws2: 0.0

Tolerances(dB):

PB: 3.0 SB: 20.0

```
*** COEFFICIENTS ***
 Filter Order = 15
0.01297
0.01859
0.02924
0.04121
0.05331
0.06412
0.07227
0.07665
0.07665
```

(a)Parks-Mc. dialog window

Script use:

Name: ParksMac

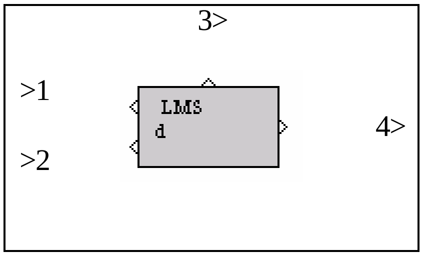
Example code: <param name = "3" value = "B0-ParksMac(3,1)">

M6.7

Block name : Least Mean Squares Algorithm **Notation**: ***LMS***

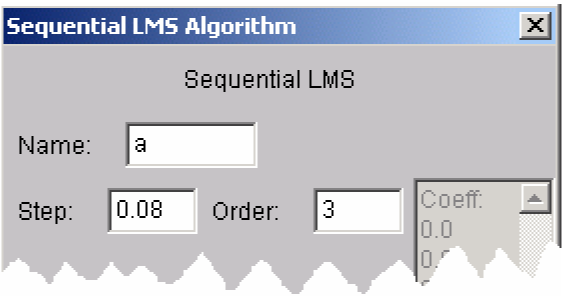
Description: Implements the sequential least mean squares adaptive filtering algorithm.

Pin assignment:



Pin	Description
1	The signal to be modeled
2	Reference signal
3	Adaptive filter coefficients
4	Adaptation error
5	
6	

Dialog window(s):



Sequential LMS

Name: a

Step: 0.08 Order: 3

Coeff:
0.0
0.

(a)LMS dialog window

Script use:

Name: LMS
Example code: <param name = “3” value = “B3-LMS(3,1)”>

Equation(s) Implemented :

A new set of adaptive filter coefficients is calculated for every new iteration in order to reduce the mean squared error. The update equation is given by

$$\boldsymbol{b}_{n+1} = \boldsymbol{b}_n + \mu e(n)\boldsymbol{x}_n$$

where $\mathbf{b}_n = \begin{bmatrix} b_0(n) \\ b_1(n) \\ \vdots \\ \vdots \\ b_{N-1}(n) \end{bmatrix}$ is the filter coefficient vector, $\mathbf{x}_n = \begin{bmatrix} x(n) \\ x(n-1) \\ \vdots \\ \vdots \\ x(n-N+1) \end{bmatrix}$, is the input vector and

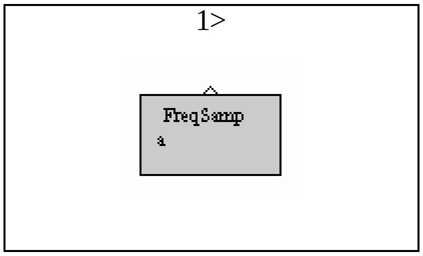
$e(n) = d(n) - \sum_{l=0}^{N-1} b_l(n)x(n-l)$ is the error signal. The step size μ is the adaptation constant that controls the rate of convergence.

M6.8

Block name : Frequency sampling **Notation**: ***FreqSamp.***

Description: This block designs a linear phase finite impulse response (FIR) filter based on the frequency sampling method. In the frequency sampling method an FIR impulse response is obtained by applying an IFFT on samples of a desired frequency response. The desired frequency response is drawn using the dialog window shown below.

Pin assignment:



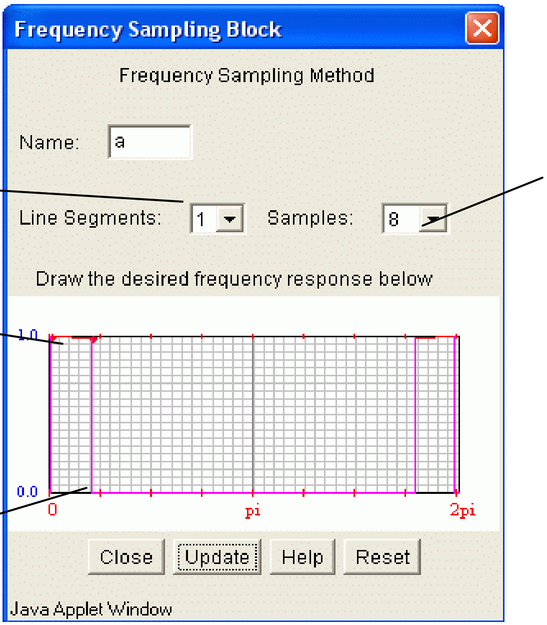
Pin	Description
1	Filter coefficients
2	
3	
4	
5	
6	

Dialog window(s):

User entry 2: Number of line segments used to draw the desired freq. response

User entry 3: Consecutive placement of points on the drawing area creates line segments related with the desired freq. response

Auxiliary lines are drawn automatically to assist the user to visualize the resulting frequency response.



Frequency Sampling Block

Frequency Sampling Method

Name: a

Line Segments: 1 Samples: 8

Draw the desired frequency response below

User entry 1: The number of samples used in the frequency sampling method. The block will export an equivalent number of FIR coefficients

(a)FreqSamp. dialog window

Script use:

Name: FreqSamp

Example code: <param name = "3" value = "B0-FreqSamp(1,7)">

Equation(s) Implemented :

$$h(n) = \frac{1}{N}\left[\sum_{k=1}^{N/2-1} 2\left|H(k)\right|\cos\left(2\pi k(n-a)/N\right) + H(0)\right]$$

$$a = (N-1)/2\,,\quad k = 0,\ldots,N-1$$

Section M5: Frequency blocks

These blocks appear at the top of the simulation area

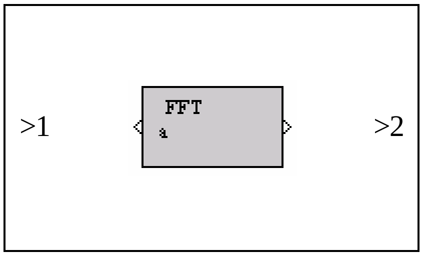
Table of blocks	
Block notation	Description
FFT	Fast Fourier Transform algorithm
IFFT	Inverse Fast Fourier Transform algorithm
Pk Pking	Peak picking routine
Magn.	Calculates the magnitude of the input signal
Phase	Calculates the phase of the input signal

M5.1

Block name : Fast Fourier Transform **Notation**: ***FFT***

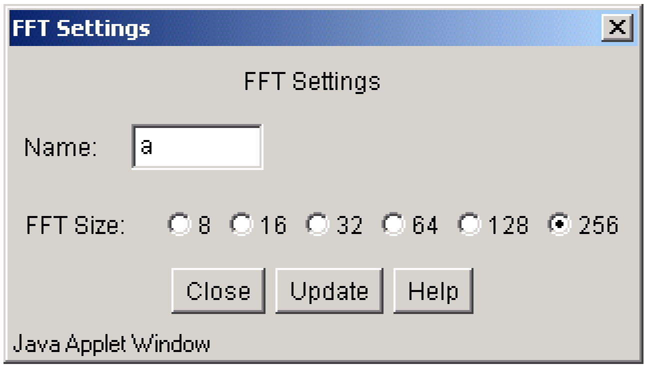
Description: Implements the Fast Fourier Transform algorithm. The user can select a desired FFT size. Possible options are 8, 16, 32, 64, 128, or 256

Pin assignment:



Pin	Description
1	Time-domain signal *x(n)*
2	Frequency-domain signal *X(k)*
3	
4	
5	
6	

Dialog window(s):


FFT Settings

FFT Settings

Name: a

FFT Size: ○ 8 ○ 16 ○ 32 ○ 64 ○ 128 ● 256

(a)FFT dialog window

Script use:

Name: fft
Example code: <param name = “3” value = “B3-fft(3,1)”>

Equation(s) Implemented :

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-j2p\,kn/N}, \qquad k = 0 \ldots N-1$$

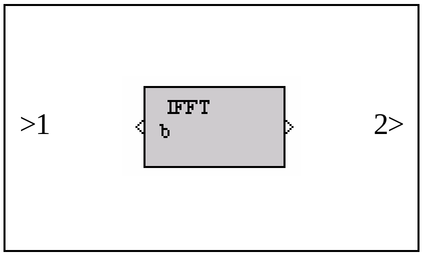
x(n) = input signal
X(k) = output signal
N = FFT length

M5.2

Block name : Inverse Fast Fourier Transform **Notation**: ***IFFT***

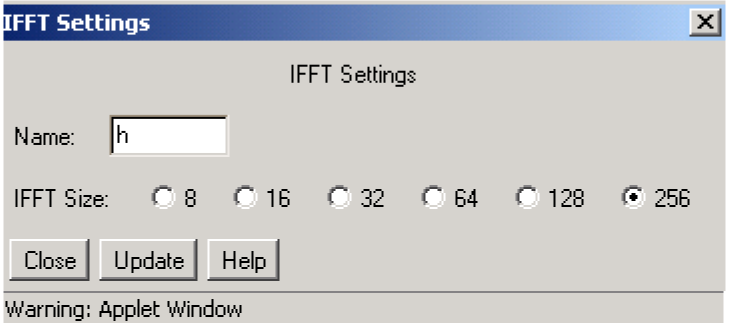
Description: Implements the Inverse Fast Fourier Transform algorithm. The user can select the desired inverse FFT size: 8, 16, 32, 64, 128, or 256.

Pin assignment:



Pin	Description
1	Frequency-domain input signal *X(k)*
2	Time-domain output signal *x(n)*
3	
4	
5	
6	

Dialog window(s):


IFFT Settings

IFFT Settings

Name: h

IFFT Size: ○ 8 ○ 16 ○ 32 ○ 64 ○ 128 ● 256

Warning: Applet Window

(a)IFFT dialog window

Script use:

Name: ifft
Example code: <param name = “3” value = “B3-ifft(3,1)”>

Equation(s) Implemented :

$$x(n) = \frac{1}{N}\sum_{k=0}^{N-1} X(k)e^{j2p\,kn/N}, \qquad n = 0 \ldots N-1$$

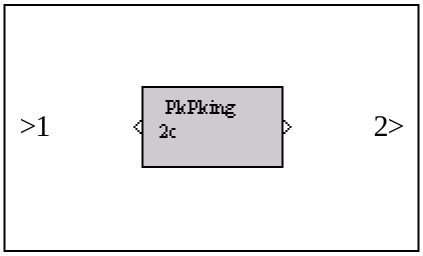
X(k) = input signal
x(n) = output signal

M5.3

Block name : Peak Picking **Notation**: ***PkPking***

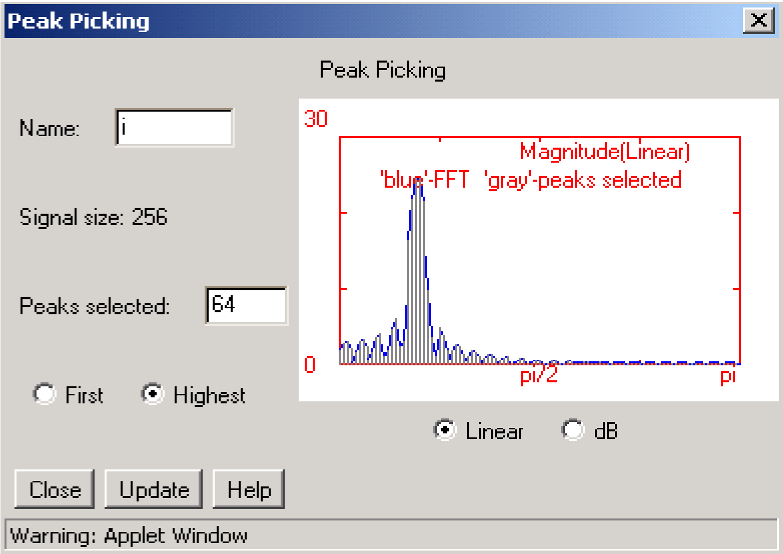
Description: Selects a specific number of peaks from a frequency-domain signal. The first set of peaks or the highest magnitude ones can be selected. Here, the "Peaks selected" option allows users to specify how many peaks to be selected. For example, 64 is chosen in the graph below. In this case, the "First" option selects the first 64 peaks of the input signal and the "Highest" option selects the 64 peaks that are the larger in magnitude.

Pin assignment:



Pin	Description
1	Frequency-domain input signal *X(k)*
2	Selected peaks
3	
4	
5	
6	

Dialog window(s):



Name: i

Signal size: 256

Peaks selected: 64

○ First ● Highest

● Linear ○ dB

Warning: Applet Window

(a)PkPking dialog window

Script use:

Name: peakpicking

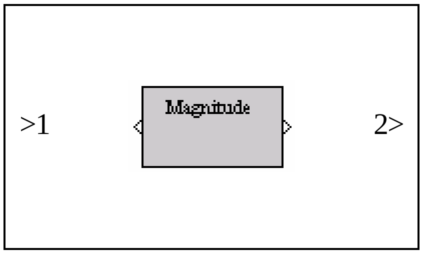
Example code: <param name = "3" value = "B3-peakpicking(3,1)">

M5.4

Block name : Magnitude **Notation**: ***Magn***

Description: This block calculates the magnitude of a signal.

Pin assignment:



Pin	Description
1	Input signal *x(n)*
2	Output signal *y(n)*
3	
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: magn

Example code: <param name = “3” value = “B3-magn(3,1)”>

Equation(s) Implemented :

$$y(n) = |\ x(n)\ |^2$$

x(n) = input signal

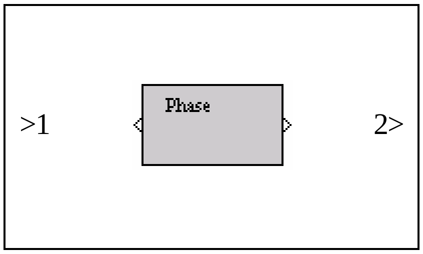
y(n) = Magnitude of the input signal

M5.5

Block name : Phase **Notation**: ***Phase***

Description: This block calculates the phase of the input signal

Pin assignment:



Pin	Description
1	Input signal
2	Phase of input signal
3	
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: phase

Example code: <param name = “3” value = “B3-phase (3,1)”>

Equation(s) Implemented :

$$f(n) = \angle x(n)$$

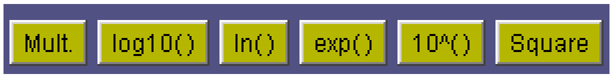
x(n) = input signal

$f(n) =$ phase of the input signal

Section M4: Arithmetic blocks

These blocks appear at the top of the simulation area

Table of blocks	
Block notation	Description
Mult.	Calculates the product of two signals
log10( )	Calculates the Log base 10 of the input signal
ln( )	Calculates the natural log of the input signal
exp( )	Calculates the exponential of the input signal
10^( )	Calculates the 10th power of the input signal
Square	Calculates the sum of squares of the two input signals

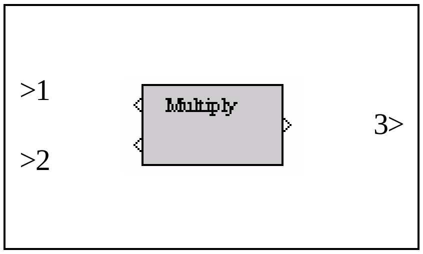


M4.1

Block name : Multip lier **Notation**: ***Mult***

Description: Multiplies the two signals at its inputs.

Pin assignment:



Pin	Description
1	Input signal $x_1(n)$
2	Input signal $x_2(n)$
3	Output signal $y(n)$
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: multiply

Example code: <param name = “3” value = “B3-multiply(3,1)”>

Equation(s) Implemented :

$$y(n) = x_1(n) \cdot x_2(n)$$

$x_1(n)$ = input signal at pin 1

$x_2(n)$ = input signal at pin 2

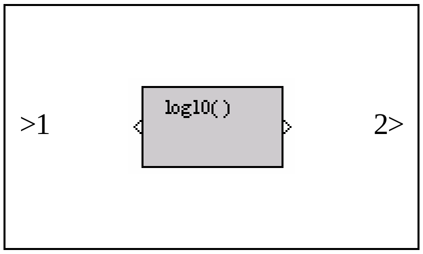
$y(n)$ = output signal

M4.2

Block name : Logarithm base 10 **Notation**: ***Log10()***

Description: This block calculates the common (base 10) logarithm of the input signal.

Pin assignment:



Pin	Description
1	Input signal*x(n)*
2	Output signal *y(n)*
3	
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: log10
Example code: <param name = “3” value = “B3-log10(3,1)”>

Equation(s) Implemented :

$$y(n) = \log_{10}(|x(n)|)$$

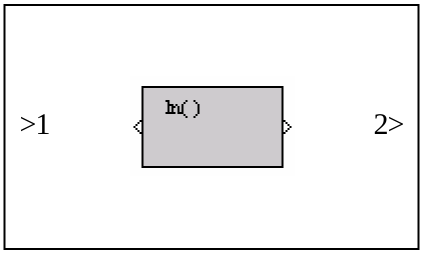
x(n) = input signal
y(n) = output signal

M4.3

Block name : Natural logarithm (base e) **Notation**: ***ln()***

Description: This block calculates the natural (base e) logarithm of the input signal.

Pin assignment:



Pin	Description
1	Input signal *x(n)*
2	Output signal *y(n)*
3	
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: ln

Example code: <param name = “3” value = “B3-ln(3,1)”>

Equation(s) Implemented :

$$y(n) = \log_e(|x(n)|)$$

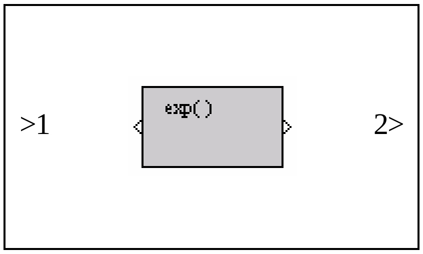
x(n) = input signal
y(n) = output signal

M4.4

Block name : Exponential **Notation**: ***exp()***

Description: This block calculates the exponential of the input signal.

Pin assignment:



Pin	Description
1	Input signal*x(n)*
2	Output signal *y(n)*
3	
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: exp
Example code: <param name = “3” value = “B3-exp(3,1)”>

Equation(s) Implemented :

$$y(n) = e^{x(n)}$$

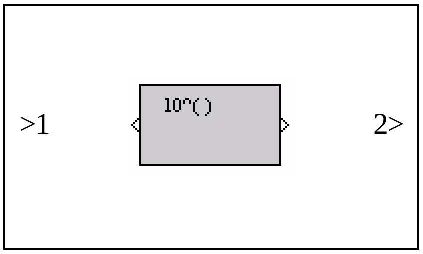
x(n) = input signal
y(n) = output signal

M4.5

Block name : Power 10 **Notation**: ***10^()***

Description: This block calculates the power 10 of the input signal

Pin assignment:



Pin	Description
1	Input signal
2	Output signal
3	
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: 10pow
Example code: <param name = “3” value = “B3-10pow(3,1)”>

Equation(s) Implemented :

$$y(n) = 10^{x(n)}$$

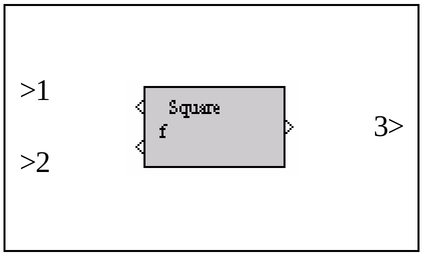
x(n) = input signal
y(n) = output signal

M4.6

Block name : Sum of squares **Notation**: ***Square***

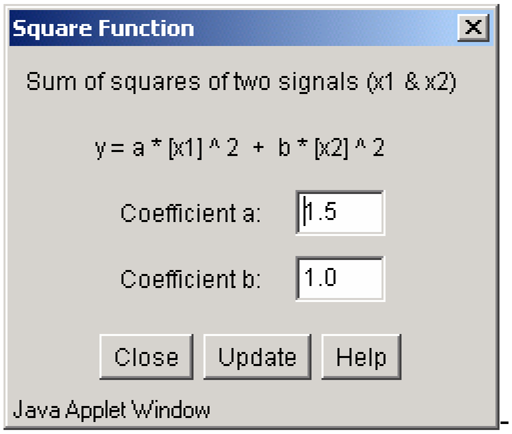
Description: This block calculates the sum of squares of the two signals at its inputs. The coefficients 'a' and 'b', are user-defined.

Pin assignment:



Pin	Description
1	Input signal $x_1(n)$
2	Input signal $x_2(n)$
3	Output signal $y(n)$
4	
5	
6	

Dialog window(s):


Square Function

Sum of squares of two signals (x1 & x2)

y = a * [x1] ^ 2 + b * [x2] ^ 2

Coefficient a: 1.5

Coefficient b: 1.0

(a)Square dialog window

Script use:

Name: square
Example code: <param name = “3” value = “B3-square(3,1)”>

Equation(s) Implemented :

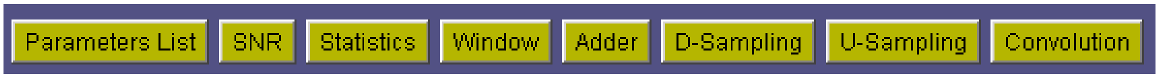
$$y(n) = ax_1^{\,2}(n) + bx_2^{\,2}(n)$$

$x_1(n)$ = input signalat pin 1
$x_2(n)$ = input signal at pin 2
$y(n)$ = output signal
a, b are the weights entered by the user

Section M3: Basic blocks

These blocks appear at the top of the simulation area

Table of blocks	
Block notation	Description
Parameters List	Lists the input parameter values
SNR	Calculates the signal-to-noise ratio between two signals
Statistics	Calculates signal statistics of the input signal
Window	Windows a time-domain signal
Mixer	Adds/subtracts two signals
D-Sampling	Down-samples a signal
U-Sampling	Up-samples a signal
Convolution	Performs convolution of two input signals

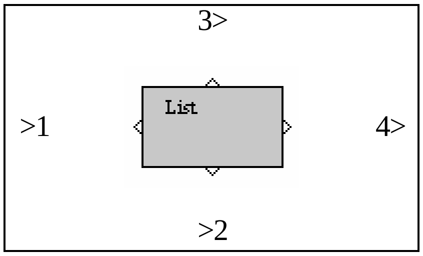


M3.1

Block name : Parameters list **Notation**: ***List***

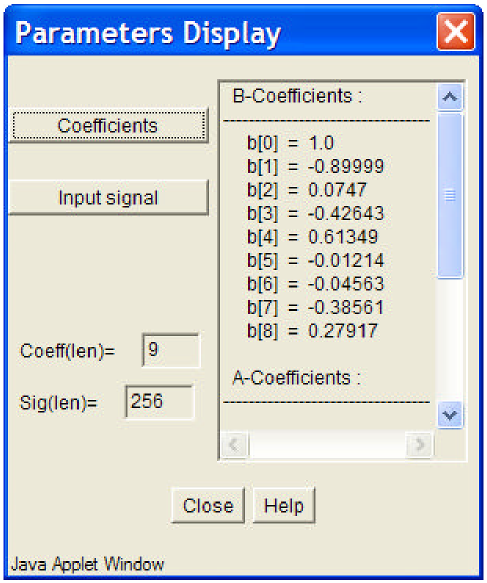
Description: This block tabulates the signal values applied at its input in a text box. No action is taken on the signals that are passed directly to the outputs. Typical signal types allowed are: filter coefficients, time domain, and frequency domain signals.

Pin assignment:



Pin	Description
1	Input signal, *x(n)*
2	Filter coefficients
3	Output coefficients
4	Output signal, *y(n)* = *x(n)*
5	
6	

Dialog window(s):



```
B-Coefficients :
------------------------------
  b[0] = 1.0
  b[1] = -0.89999
  b[2] = 0.0747
  b[3] = -0.42643
  b[4] = 0.61349
  b[5] = -0.01214
  b[6] = -0.04563
  b[7] = -0.38561
  b[8] = 0.27917

A-Coefficients :
------------------------------
```

Coeff(len)= 9

Sig(len)= 256

(a)List dialog window

Script use:

Name: list

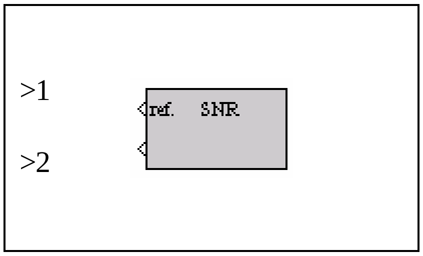
Example code: <param name = “3” value = “B3-list(3,1)”>

M3.2

Block name : Signal-to-noise ratio **Notation**: ***SNR***

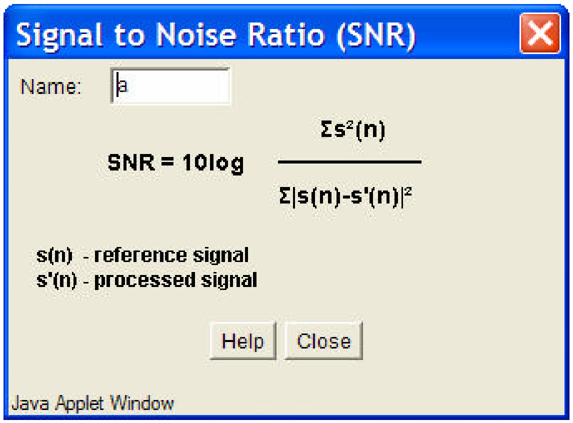
Description: This block calculates the signal-to-noise ratio (SNR) value in 'dB' between two signals. The reference signal is given as input to the upper input pin.

Pin assignment:



Pin	Description
1	Reference signal
2	Processed signal
3	
4	
5	
6	

Dialog window(s):


Signal to Noise Ratio (SNR)

Name: a

$$SNR = 10\log \frac{\Sigma s^2(n)}{\Sigma|s(n)-s'(n)|^2}$$

s(n) - reference signal
s'(n) - processed signal

(a)SNR dialog window

Script use:

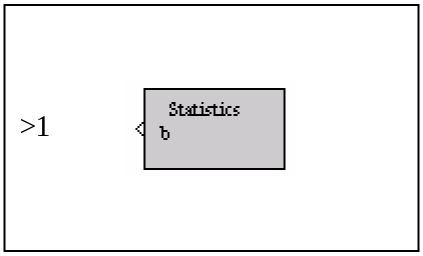
Name: snr
Example code: <param name = "3" value = "B3-snr(3,1)">

M3.3

Block name : Statistics **Notation**: ***Statistics***

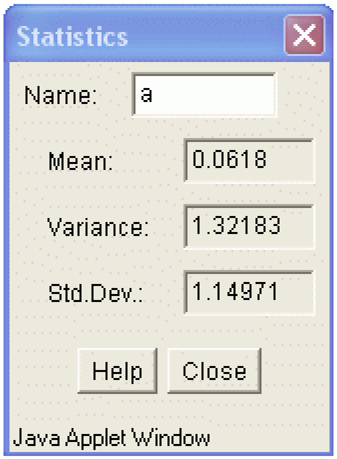
Description: This block computes the first-order statistics of the input signal i.e. the mean, the variance, and the standard deviation. The mean is calculated as the sum of the individual samples of the input, divided by the number of samples. The variance is a measure of the deviation from the mean. Standard deviation is the square root of the variance.

Pin assignment:



Pin	Description
1	Input signal
2	
3	
4	
5	
6	

Dialog window(s):



Name: a

Mean: 0.0618

Variance: 1.32183

Std.Dev.: 1.14971

(a)Statistics dialog window

Script use:

Name: stats

Example code: <param name = “3” value = “B3-stats(3,1)”>

Equation(s) Implemented :

Mean: $m_x = \frac{1}{N}\sum_{n=1}^{N} x(n)$, Variance: $s_x^2 = \frac{1}{N}\sum_{n=1}^{N}(x(n) - m_x)^2$, Standard deviation $= s_x$

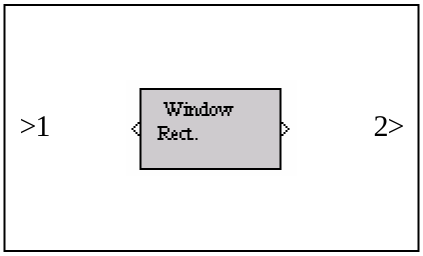
x(n) = input signal of length *N*

M3.4

Block name : Window **Notation**: ***Window***

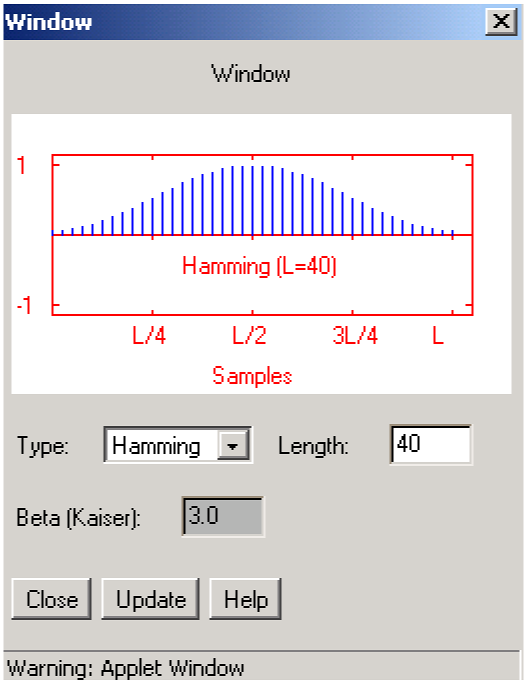
Description: This block performs a windowing operation on the input signal The available window functions are: Hamming, Hanning, rectangular, Bartlett, Blackman, and Kaiser. The maximum window length is 256 samples.

Pin assignment:



Pin	Description
1	Input signal, *x(n)*
2	Windowed signal, *y(n)*
3	
4	
5	
6	

Dialog window(s):



Type: Hamming

Length: 40

Beta (Kaiser): 3.0

Warning: Applet Window

(a) Window dialog window

Script use:

Name: window

Example code: <param name = “3” value = “B3-window(3,1)”>

Equation(s) Implemented :

$$y(n) = w(n)x(n)$$

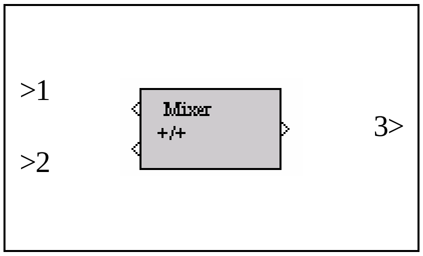
x(n) = input signal
w(n) = windowing function
y(n) = windowed signal

M3.5

Block name : Mixer (or Adder) **Notation**: ***Mixer***

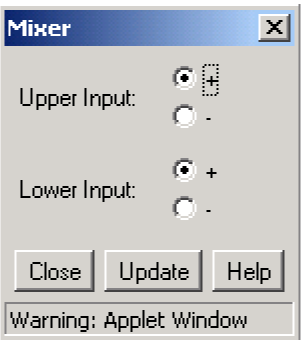
Description: Adds or subtracts two signals

Pin assignment:



Pin	Description
1	Input signal $x_1(n)$
2	Input signal $x_2(n)$
3	Output signal $y(n)$
4	
5	
6	

Dialog window(s):



Upper Input: ● + ○ -

Lower Input: ● + ○ -

Warning: Applet Window

(a)Mixer dialog window

Script use:

Name: mixer
Example code: <param name = “3” value = “B3-mixer(3,1)”>

Equation(s) Implemented :

$$y(n) = x_1(n) \pm x_2(n)$$

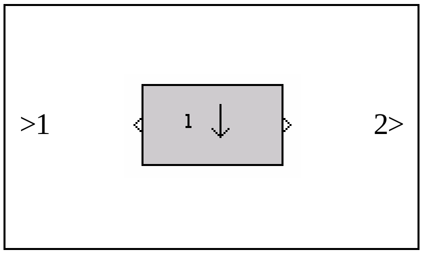
$x_1(n)$ = input signal at pin 1
$x_2(n)$ = input signal at pin 2
$y(n)$ = output signal

M3.6

Block name : Down-sampling **Notation**: ***D-Sampling***

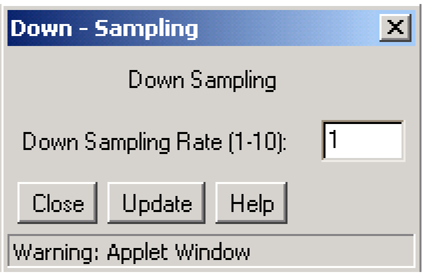
Description: Down–samples the input signal by an integer factor M

Pin assignment:



Pin	Description
1	Input signal
2	Down-sampled signal
3	
4	
5	
6	

Dialog window(s):



Down Sampling

Down Sampling Rate (1-10): 1

Warning: Applet Window

(a)D-Sampling dialog window

Script use:

Name: dsample
Example code: <param name = “3” value = “B3-dsample(3,1)”>

Equation(s) Implemented :

$$y(n) = x(nM)$$

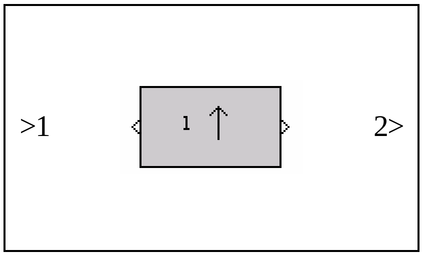
x(n) = input signal
y(n) = output signal
M = down-sampling factor

M3.7

Block name : Up-sampling **Notation**: ***U-Sampling***

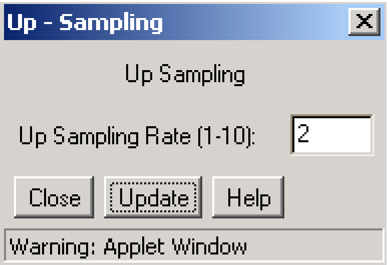
Description: Up-samples the input signal by an integer factor *L*. *L* is allowed to take values from 1 to 10.

Pin assignment:



Pin	Description
1	Input signal
2	Up-sampled signal
3	
4	
5	
6	

Dialog window(s):



Up Sampling Rate (1-10): 2

(a)U-Sampling dialog window

Script use:

Name: usample

Example code: <param name = “3” value = “B3-usample(3,1)”>

Equation(s) Implemented :

$$y(n) = x(n / L)$$

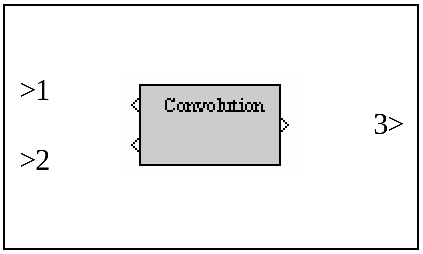
x(n) = input signal
y(n) = output signal
L = up-sampling factor

M3.8

Block name : Convolution **Notation**: ***Convolution***

Description: This block performs a convolution operation between its input signals.

Pin assignment:



Pin	Description
1	Input signal *x1(n)*
2	Input signal *x2(n)*
3	Convolved signal *y(n)*
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: conv
Example code: <param name = “3” value = “B3-conv(3,1)”>

Equation(s) Implemented :

$$y(n) = x_1(n) * x_2(n), \qquad y(n) = \sum_{m=0}^{N-1} x_1(m)\, x_2(n-m)$$

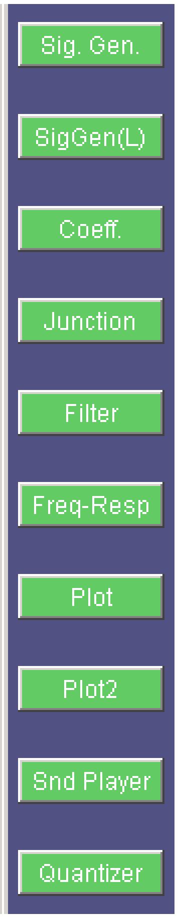
x1(n) = input signal
x2(n) = input signal
y(n) = convolved signal

Section M2: General blocks

These blocks appear at the left of the simulation area

Table of blocks	
Block notation	Description
Sig Gen.	Generates signals of length up to 256 samples
Sig Gen (L)	Generates signals of more than 256 samples
Coeff.	Allows entering numerator/denominator coefficients for filters
Junction	Routes its input to its two outputs
Filter	Filters input signal based on provided coefficients
Freq.Resp	Calculates and displays frequency response of a filter
Plot	Plots a single signal.
Plot2	Plots two signals for comparison purposes
Snd Player	Performs signal playback
Quantizer	Performs signal quantization

- See next page for button diagram -

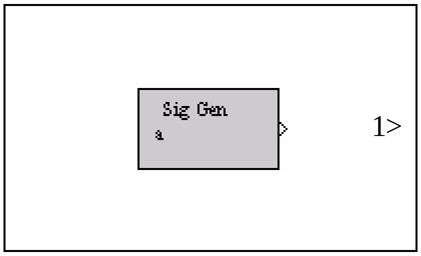

Sig. Gen.
SigGen(L)
Coeff.
Junction
Filter
Freq-Resp
Plot
Plot2
Snd Player
Quantizer

M2.1

Block name : Signal generator **Notation**: ***SigGen***

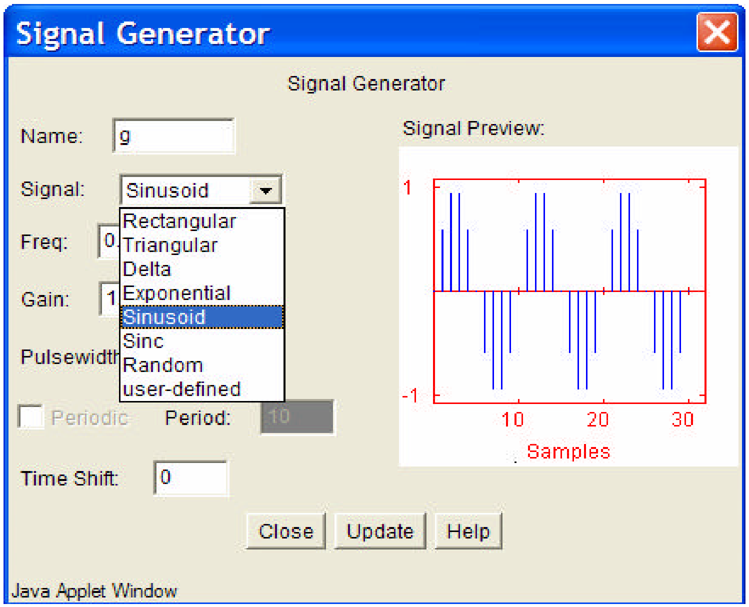
Description: Generates a variety of time-domain signals. It supports: pulses, triangular, delta, exponential, sinusoid, sinc, random, and user-defined signals. The length of each signal ("pulse width") and the amplitude of the signal ("gain") can be set. A signal can be made periodic if the "periodic" option is selected. The base of the exponential can also be varied. Random signals can have uniform, normal, and Rayleigh distributions with variable mean and variance.

Pin assignment:



Pin	Description
1	Time-domain signal
2	
3	
4	
5	
6	

Dialog window(s):


Signal Generator

Signal Generator

Name: g

Signal: Sinusoid

Freq: 0.

Gain: 1

Pulsewidth

□ Periodic Period: 10

Time Shift: 0

Java Applet Window

(a)SigGen dialog window

Script use:

Name: siggen

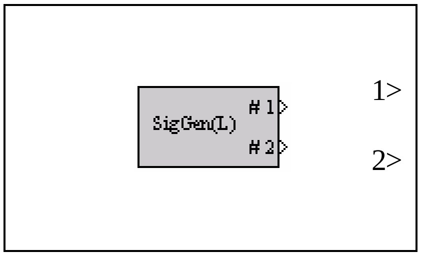
Example code: <param name = "3" value = "B3-siggen(3,1)">

M2.2

Block name : Long signal generator **Notation**: ***SigGen(L)***

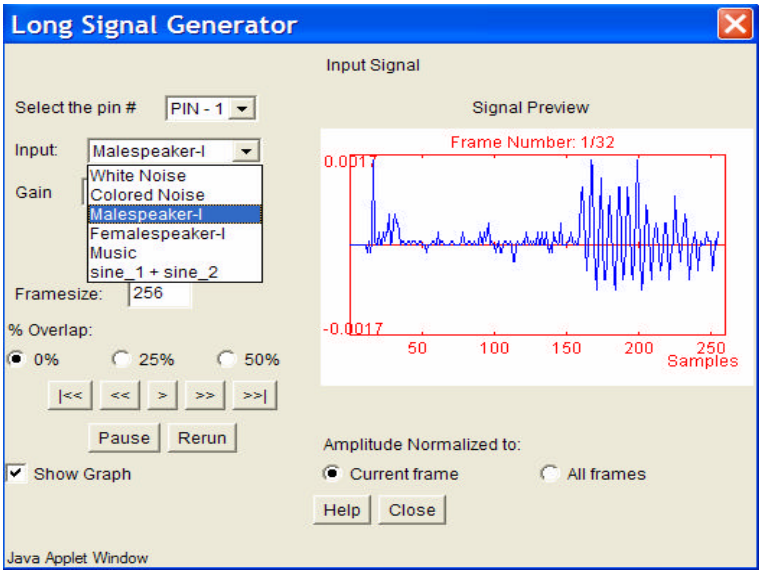
Description: This block produces 6 types of signals, i.e., male speech, female speech, music, white noise, colored noise, and sinusoid with a maximum data length of 8192 samples. The sinusoid option can generate a sum of two sinusoids, based on the specified frequencies and amplitudes. I desired, an option is provided to synchronize the part's two independent outputs. The option "frame size" represents the number of samples in each frame. The option "overlap" allows frames to overlap. Possible overlapping schemes are: 0%, 25% and 50%. The output plot may be displayed with the signal normalized either with respect to the maximum magnitude of the current frame or the maximum of the entire signal. When the colored noise signal is selected, a new window is created where filter coefficients that convert white noise to colored noise can be entered. The frames can be directed to the output individually *(">>")* or all together automatically *(">>|")*.

Pin assignment:



Pin	Description
1	Time-domain signal 1 (in frames)
2	Time-domain signal 2 (in frames)
3	
4	
5	
6	

Dialog window(s):



Select the pin # PIN - 1

Input: Malespeaker-I

Gain

Framesize: 256

% Overlap: ● 0% ○ 25% ○ 50%

☑ Show Graph

Amplitude Normalized to: ● Current frame ○ All frames

(a)SigGen(L) dialog window

Script use:

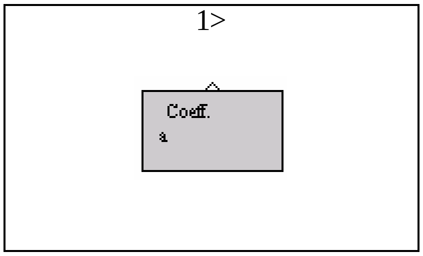
Name: siggen(L)
Example code: <param name = "3" value = "B3-siggen(L)(3,1)">

M2.3

Block name : Coefficient **Notation**: ***Coeff***

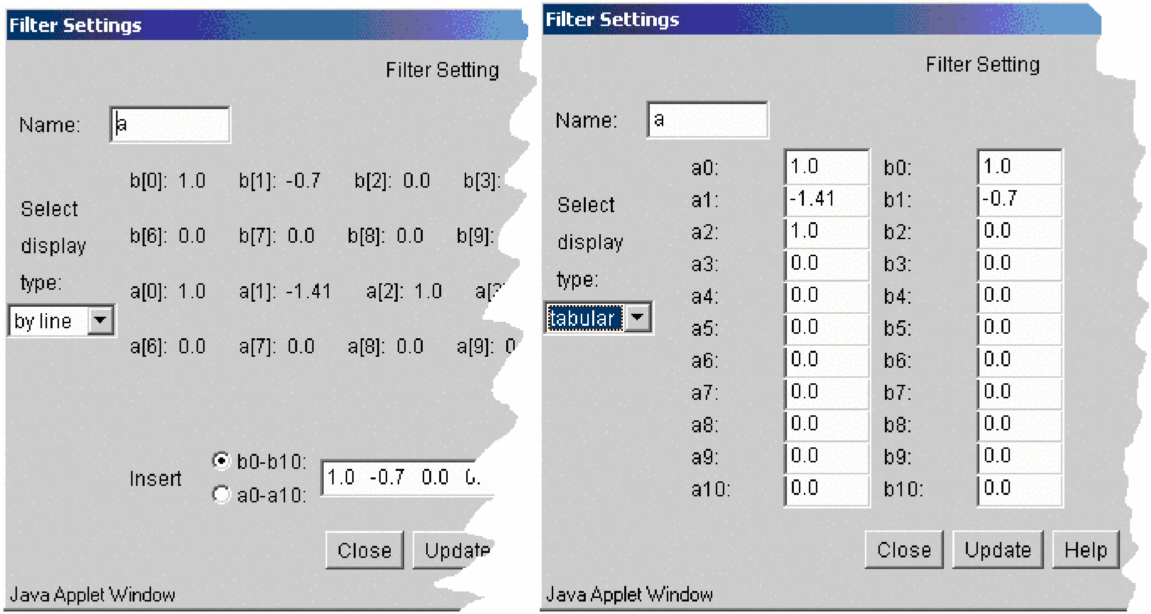
Description: This block allows the user to enter filter coefficients. A maximum of 11 coefficients can be used. Coefficients can be entered in "tabular" form or "by line" form as shown below. The "by line" option provides an easy way to 'cut' and 'paste' coefficients from other sources.

Pin assignment:



Pin	Description
1	Coefficients (numerator and denominator)
2	
3	
4	
5	
6	

Dialog window(s):



Name: a

b[0]: 1.0 b[1]: -0.7 b[2]: 0.0 b[3]:

b[6]: 0.0 b[7]: 0.0 b[8]: 0.0 b[9]:

a[0]: 1.0 a[1]: -1.41 a[2]: 1.0 a[

a[6]: 0.0 a[7]: 0.0 a[8]: 0.0 a[9]: 0

Select display type: by line

Insert ● b0-b10: 1.0 -0.7 0.0 0.
○ a0-a10:

Name: a

Select display type: tabular

a0:	1.0	b0:	1.0
a1:	-1.41	b1:	-0.7
a2:	1.0	b2:	0.0
a3:	0.0	b3:	0.0
a4:	0.0	b4:	0.0
a5:	0.0	b5:	0.0
a6:	0.0	b6:	0.0
a7:	0.0	b7:	0.0
a8:	0.0	b8:	0.0
a9:	0.0	b9:	0.0
a10:	0.0	b10:	0.0

(a)Coefficient dialog window-by line and tabular

Script use:

Name: Coeff
Example code: <param name = "3" value = "B3-coeff(3,1)">

Equation(s) Implemented :

$$y(n) = \sum_{i=0}^{L} b_i x(n-i) - \sum_{i=1}^{M} a_i y(n-i)$$

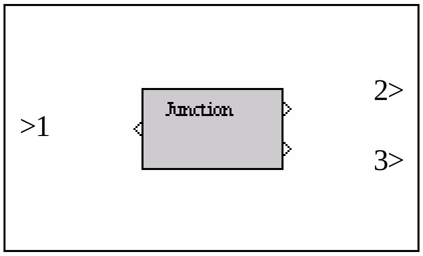
x(n) = input signal, *y(n)* = output signal, a_i = feedback coefficients, b_i = feed-forward coefficients

M2.4

Block name : Junction **Notation**: ***Junction***

Description: This block propagates its input signal at its two outputs. The input signal can be either time-domain, frequency-domain, or filter coefficients. The ***Junction*** block essentially allows other blocks to share the same signal or parameters

Pin assignment:



Pin	Description
1	Input signal
2	Output signal = input signal
3	Output signal = input signal
4	
5	
6	

Dialog window(s):

-None-

Script use:

Name: junction
Example code: <param name = “3” value = “B3-junction(3,1)”>

Equation(s) Implemented :

$$x(n) = y(n) = z(n)$$

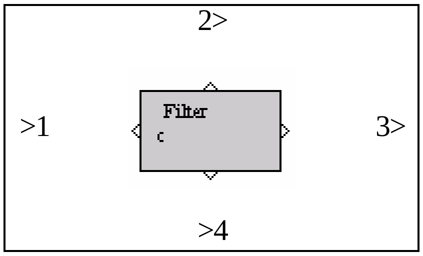
x(n) = input signal
y(n) = output signal at first output pin
z(n) = output signal at second output pin

M2.5

Block name: Filter **Notation**: ***Filter***

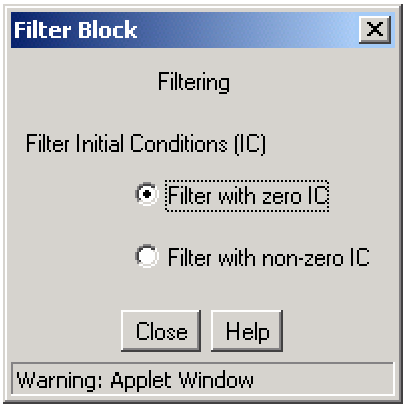
Description: This block filters the input signal based on the provided numerator and denominator coefficients and the standard difference equation. The filter coefficients must be provided using the ***Coeff.*** block. An option is provided to start with zero initial conditions or non-zero initial conditions.

Pin assignment:



Pin	Description
1	Input signal *x(n)*
2	Filter coefficients
3	Filtered signal *y(n)*
4	Feedback and feed-forward coefficients a_i and b_i
5	
6	

Dialog window(s):



Filter Block

Filtering

Filter Initial Conditions (IC)

● Filter with zero IC

○ Filter with non-zero IC

Warning: Applet Window

(a)Filter dialog window

Script use:

Name: filter

Example code: <param name = “3” value = “B3-filter(3,1)”>

Equation(s) Implemented :

$$y(n) = \sum_{i=0}^{L} b_i x(n-i) - \sum_{i=1}^{M} a_i y(n-i)$$

x(n) = input signal

y(n) = output signal

a_i = feedback coefficients

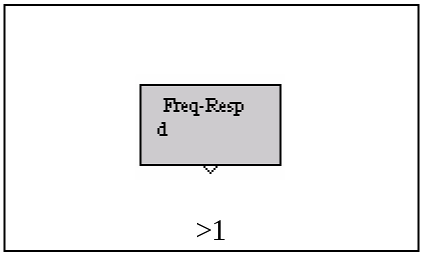
b_i = feed-forward coefficients

M2.6

Block name : Frequency response **Notation**: ***Freq-Resp***

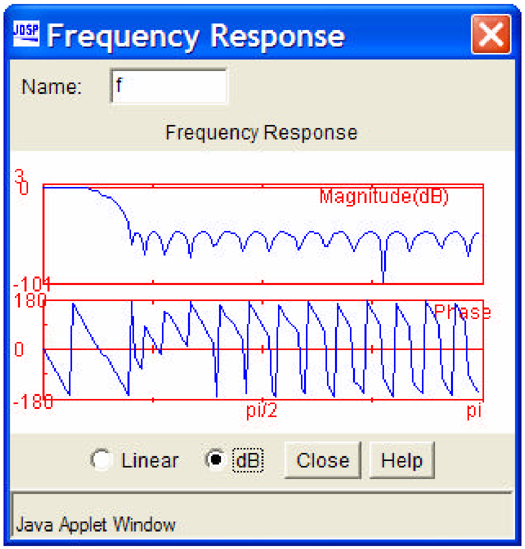
Description: This block calculates and displays the frequency response of a filter. It can be connected to any block that can generate filter coefficients. In its dialog window, the top plot displays the magnitude in dB or linear scale and the bottom plot shows the phase.

Pin assignment:



Pin	Description
1	Feedback and feed-forward coefficients a_i and b_i
2	
3	
4	
5	
6	

Dialog window(s):



Name: f

○ Linear ● dB

(a)Frequency Response dialog window

Script use:

Name: freqresp

Example code: <param name = “3” value = “B3-freqresp(3,1)”>

Equation(s) Implemented :

$$H(e^{j\Omega}) = \frac{\sum_{i=0}^{L} b_i e^{-ji\Omega}}{1 + \sum_{i=1}^{M} a_i e^{-ji\Omega}}$$

a_i = feedback coefficients

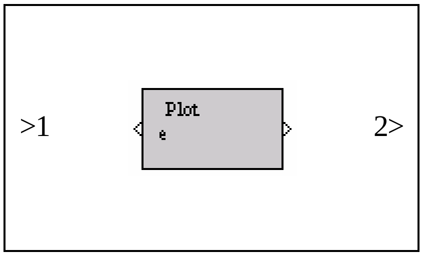
b_i = feed-forward coefficients

M2.7

Block name : Plot **Notation**: ***Plot***

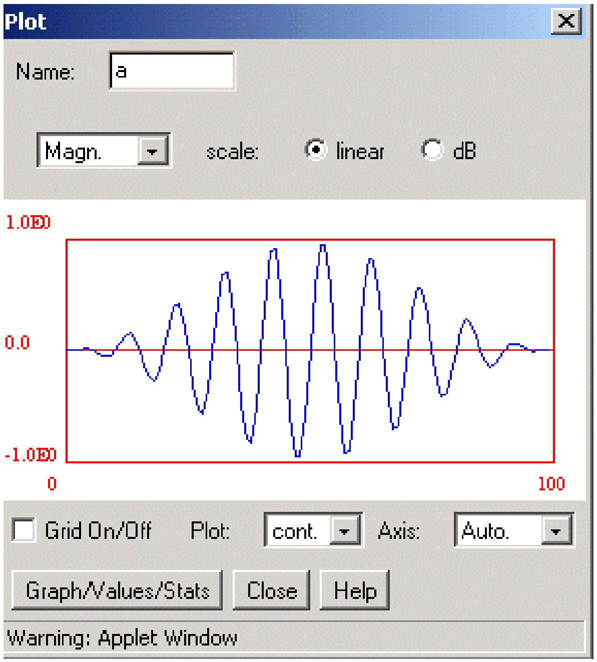
Description: This block primarily plots the signal at its input in an x-y axis coordinate system. It can also display values in text form and calculate some basic signal statistics. The magnitude, magnitude squared, real part, imaginary part, and phase of the input signal can be examined.

Pin assignment:



Pin	Description
1	Input signal *x(n)*
2	Output signal *y(n)* = input signal *x(n)*
3	
4	
5	
6	

Dialog window(s):



Plot

Name: a

Magn. scale: ● linear ○ dB

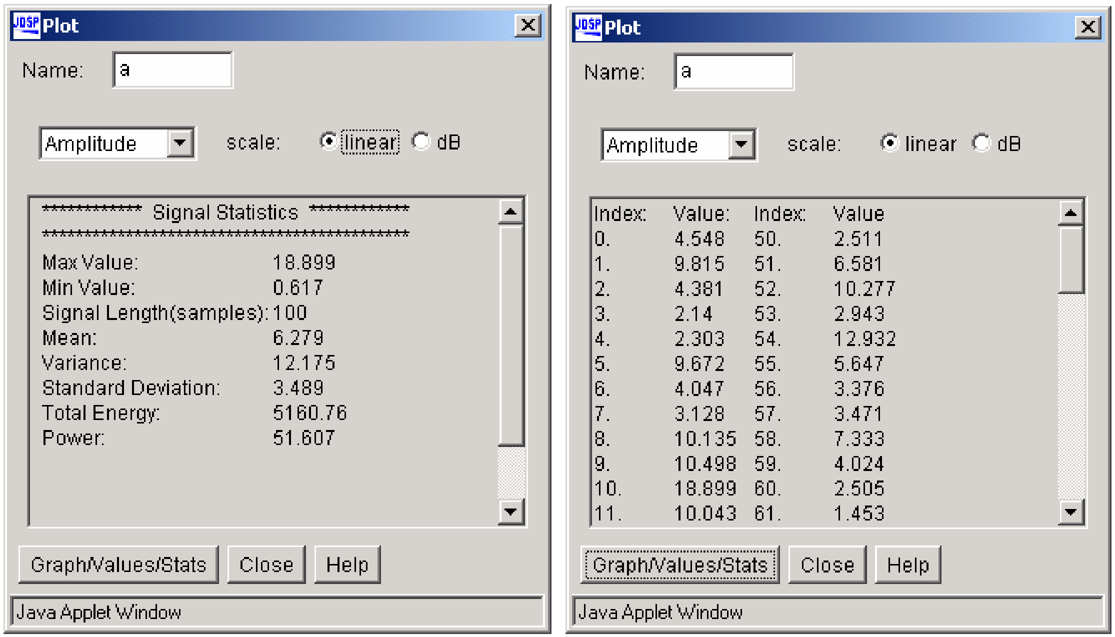
□ Grid On/Off Plot: cont. Axis: Auto.

Warning: Applet Window

Plot tools:

- Grid
- Graphical zooming
- Manual adjustment of axes
- Displaying the signal as continuous or discrete
- dB/linear scale for magnitude plots and deg./rad for phase plots
- Time-domain signals are plotted in terms of time samples
- Frequency-domain signals are plotted in terms of radians

(a)Plot dialog window-Graphical



Name: a

Amplitude scale: ● linear ○ dB

```
************ Signal Statistics ************
*******************************************
Max Value:                18.899
Min Value:                0.617
Signal Length(samples):100
Mean:                     6.279
Variance:                 12.175
Standard Deviation:       3.489
Total Energy:             5160.76
Power:                    51.607
```

Name: a

Amplitude scale: ● linear ○ dB

```
Index:  Value:   Index:  Value
0.      4.548    50.     2.511
1.      9.815    51.     6.581
2.      4.381    52.     10.277
3.      2.14     53.     2.943
4.      2.303    54.     12.932
5.      9.672    55.     5.647
6.      4.047    56.     3.376
7.      3.128    57.     3.471
8.      10.135   58.     7.333
9.      10.498   59.     4.024
10.     18.899   60.     2.505
11.     10.043   61.     1.453
```

(b)Plot dialog window-Statistics and values

Script use:

Name: plot
Example code: <param name = “3” value = “B3-plot(3,1)”>

Equation(s) Implemented :

$$\text{Mean}, m_x = \frac{1}{N}\sum_{n=1}^{N} x(n) \qquad \text{Variance}, s_x^2 = \frac{1}{N}\sum_{n=1}^{N} (x(n) - m_x)^2$$

$$\text{Standard Deviation} = s_x \qquad \text{Total energy} = \sum_{n=1}^{N} x^2(n) \qquad \text{Power} = \frac{1}{N}\sum_{n=1}^{N} x^2(n)$$

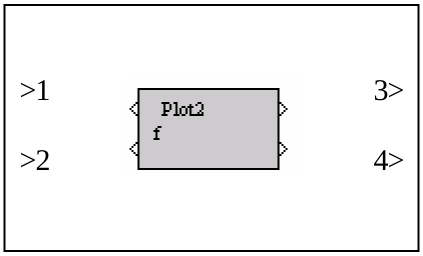
x(n) = input signal, *N* = number of samples

M2.8

Block name : Plot 2 **Notation**: ***Plot2***

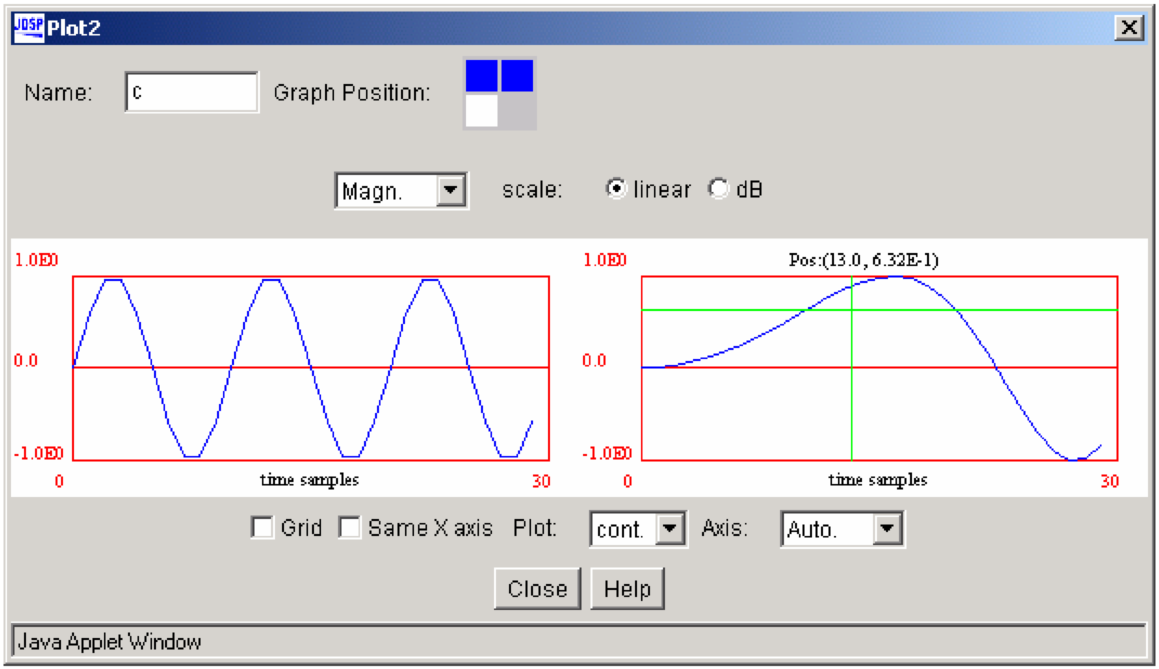
Description: Plots two signals in the same dialog window. All signals are plotted in terms of samples, and any scale changes apply to both graphs. Graphs can be plotted one below the other, one next to the other or in the same axis. Use the "Graph Position" option to vary the graph location.

Pin assignment:



Pin	Description
1	Input signal *x(n)*
2	Input signal y*(n)*
3	Output signal *z(n)* = Input signal *x(n)*
4	Output signal g*(n)* = Input signal y*(n)*
5	
6	

Dialog window(s):



Name: c

Graph Position:

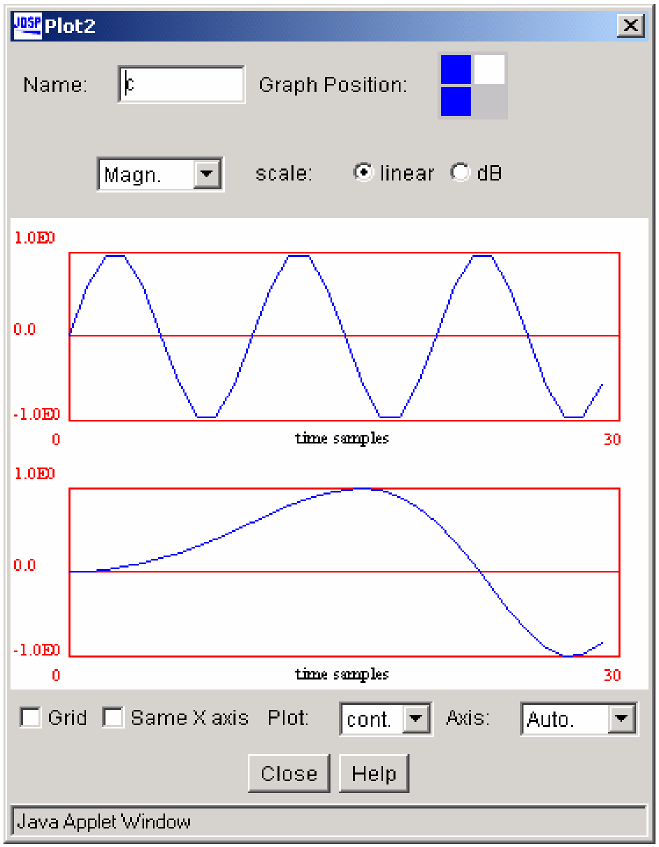
Magn.

scale: ● linear ○ dB

□ Grid □ Same X axis Plot: cont. Axis: Auto.

Java Applet Window

(a)Plot 2 dialog window-Horizontal orientation



Name: c

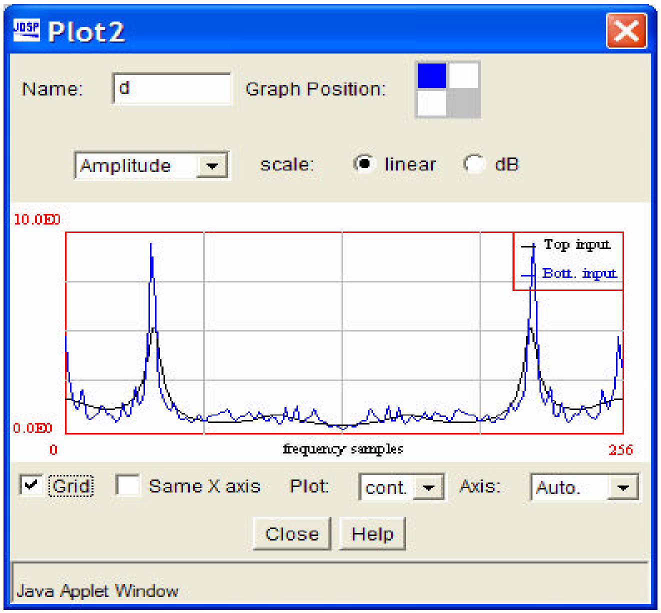
Graph Position:

Magn.

scale: ● linear ○ dB

□ Grid □ Same X axis Plot: cont. Axis: Auto.

(b)Plot 2 dialog window-Vertical orientation



Name: d

Graph Position:

Amplitude

scale: ● linear ○ dB

☑ Grid □ Same X axis Plot: cont. Axis: Auto.

(c)Plot 2 dialog window- same axis option

Script use:

Name: plot2

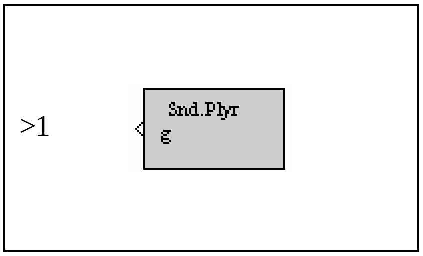
Example code: <param name = “3” value = “B3-plot2(3,1)”>

M2.9

Block name : Sound Player **Notation**: ***SndPlyr***

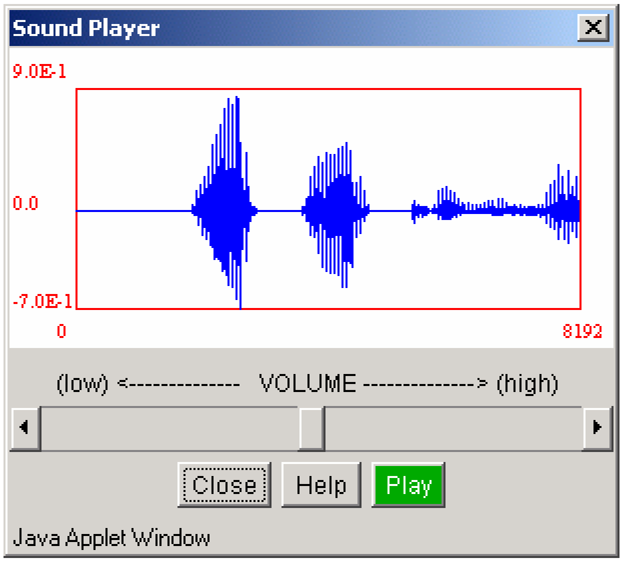
Description: This block is used for signal playback. Dragging the volume scroll bar to the right increases the signal volume.

Pin assignment:



Pin	Description
1	Input signal *x(n)*
2	
3	
4	
5	
6	

Dialog window(s):



(a)Sound Plyr dialog window

Script use:

Name: sndplayer

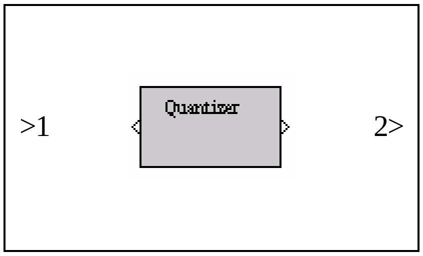
Example code: <param name = “3” value = “B3-sndplayer(3,1)”>

M2.10

Block name : Quantizer **Notation**: ***Quantizer***

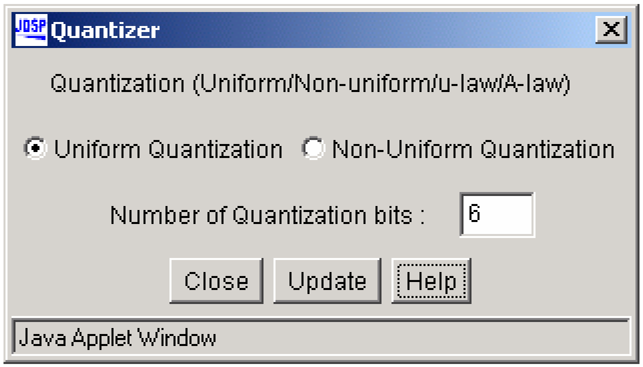
Description: This block is used for signal quantization. Uniform or non-uniform quantization can be selected. For uniform quantization, the amplitude levels are divided into steps of $(0.5)^n$, where n is the number of quantization bits. These discrete levels are used to represent the signal amplitudes. Non-uniform quantization is achieved by uniformly quantizing a µ-law or A-law compressed signal. Note that this block can only simulate the effect of quantization on signals or on filter coefficients.

Pin assignment:



Pin	Description
1	Input signal *x(n)*
2	Quantized signal y*(n)*
3	
4	
5	
6	

Dialog window(s):



Quantization (Uniform/Non-uniform/u-law/A-law)

● Uniform Quantization ○ Non-Uniform Quantization

Number of Quantization bits : 6

Java Applet Window

(a)Quantizer dialog window

Script use:

Name: quant
Example code: <param name = “3” value = “B3-quant (3,1)”>

Equation(s) Implemented :

µ– law can be stated as, $x_{out} = \dfrac{\log(1+m|x_{in}|)}{\log(1+m)}$; µ = 0;

A– law can be stated as, $x_{out} = \dfrac{A\,|x_{in}|}{1+\log A}$; $0 = x_{in} = 1/A$ and $x_{out} = \dfrac{1+\log(A\,|x_{in}|)}{1+\log A}$; $1/A = x_{in} = 1$

where, x_{in} and x_{out} are the normalized input and output signal amplitudes and $A = 1$

Section M1 Introduction

1.1. General Information on J-DSP

J-DSP is an object-oriented Java™ tool, where J-DSP stands for Java Digital Signal Processing. J-DSP has been developed at Arizona State University (ASU) and is written as a platform-independent Java applet that resides either on a server or on a local hard-drive. It is accessible through the use of a web browser. J-DSP has a rich suite of signal processing functions that facilitate interactive on-line simulations of modern statistical signal and spectral analysis algorithms filter design tools, QMF banks, and state-of-the-art vocoders.

All functions in J-DSP appear as graphical blocks that are divided into groups according to their functionality. Selecting and establishing individual blocks is done using a drag-and-drop-process. Each block is linked to a signal processing function. Fig. 1 shows the J-DSP editor environment and Fig. 2 shows details on the drop-down menus and the signal processing functions of J-DSP. A simulation can be started by connecting appropriate blocks from left to right. Signals at any point of a simulation can be analyzed and plotted through the use of appropriate functions. Parameters in the blocks can be edited through dialog windows. Blocks can easily be manipulated (edit, move, delete and connect) using the mouse. Execution is dynamic and therefore any change at any point of a simulated system will automatically take effect in all related blocks. All dialog windows can be left open to enable viewing results at more than one point in the editor.

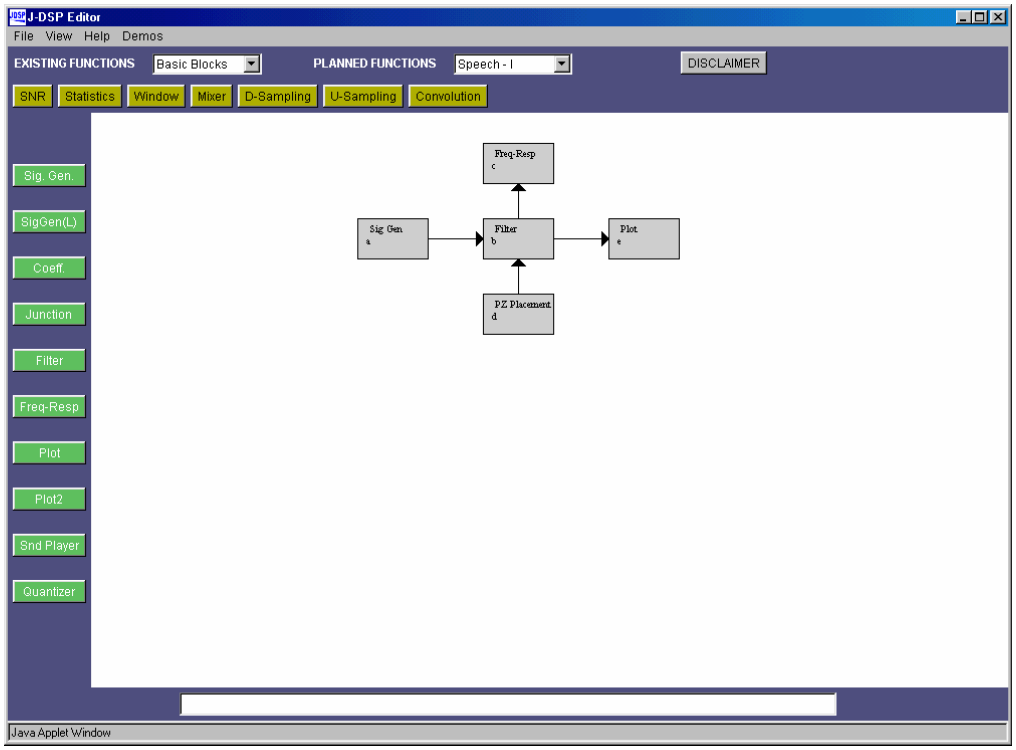
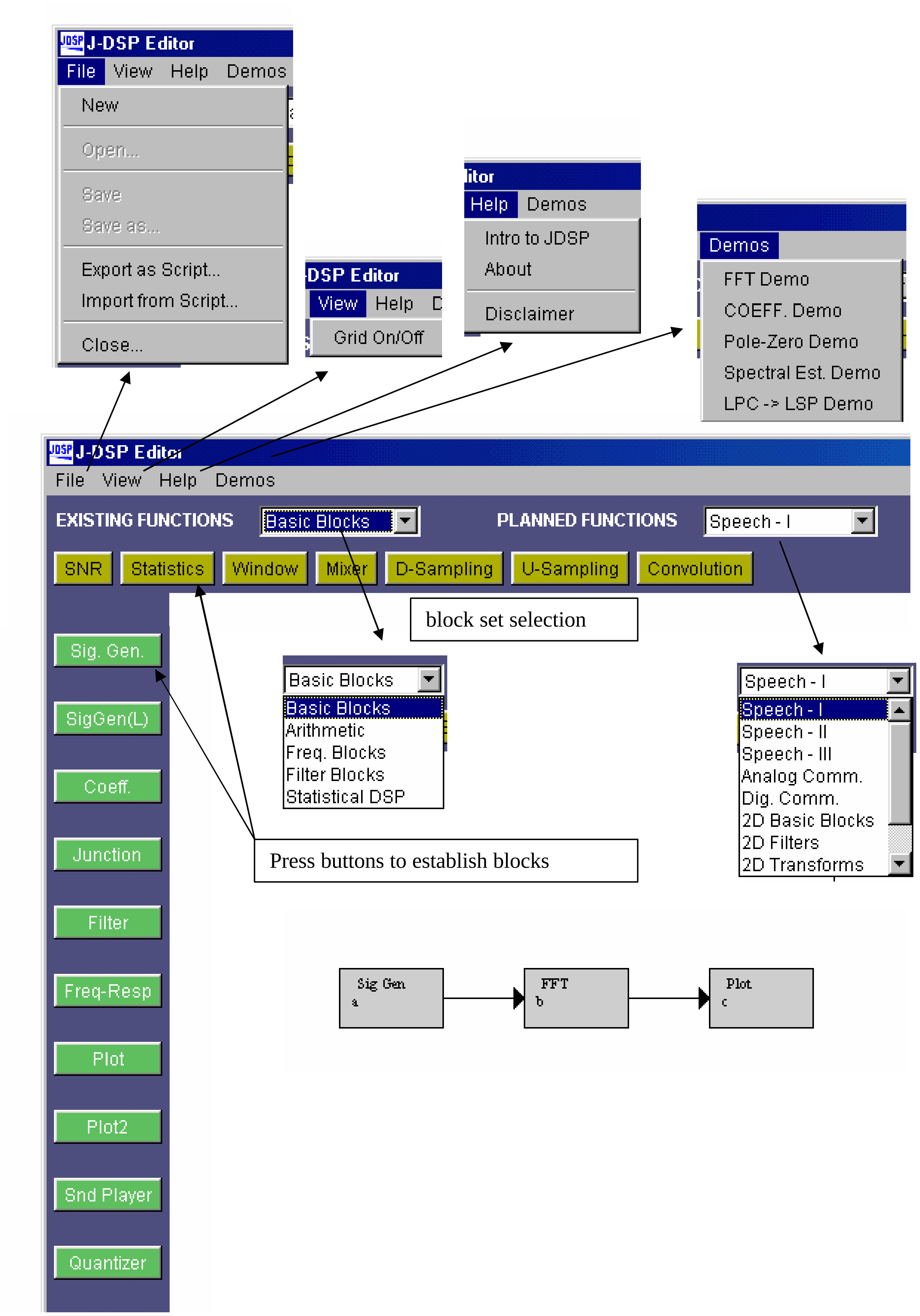


Fig 1. J-DSP simulation environment



block set selection

Press buttons to establish blocks

Fig. 2. Signal processing functions and menus in J-DSP

1.2. Working with J-DSP

The easiest way to explain some of the functions of J-DSP is to work through a simple example. To start J-DSP, go to the link http://jdsp.asu.edu/, click on “Start J-DSP”, and press [Start] in the subsequent dialog window. Then if you agree, press the [I Accept] button at the disclaimer window. A relatively large Java applet will start downloading to the browser. Even though the J-DSP working area appears almost immediately, it may take 30 seconds or more to establish the first block for a telephone-based (28.8) Internet connection. However, once the first block is established the program should run reasonably fast. Adjust the size of the J-DSP editor window so that you are able to view the entire work area. Press the SigGen button on the left part of the window. Move the mouse to the center of the window and click the left mouse button. Note that you have created the signal generator block. There are two signal generators, ***SigGen*** for processing a single frame of the signal and ***SigGen(L)*** for frame-by-frame processing that is typically used in speech processing simulations. Similarly, create a ***Filter*** and a ***Plot*** block as shown in Fig. 3. Note that blocks cannot be placed on top of one another. There are two plot blocks, i.e., ***Plot*** (single plot) and ***Plot2*** (two plots). For now, use ***Plot***.

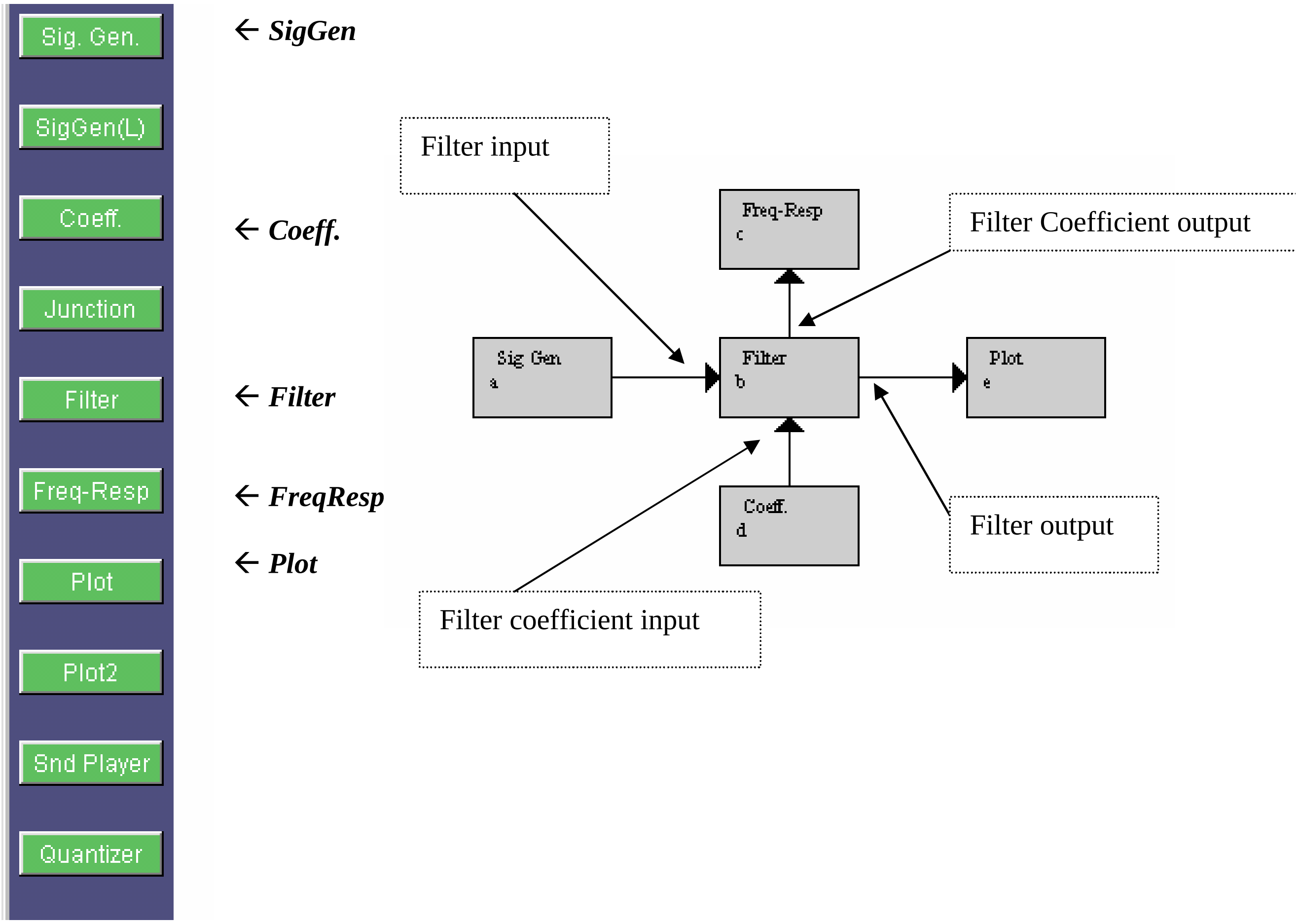
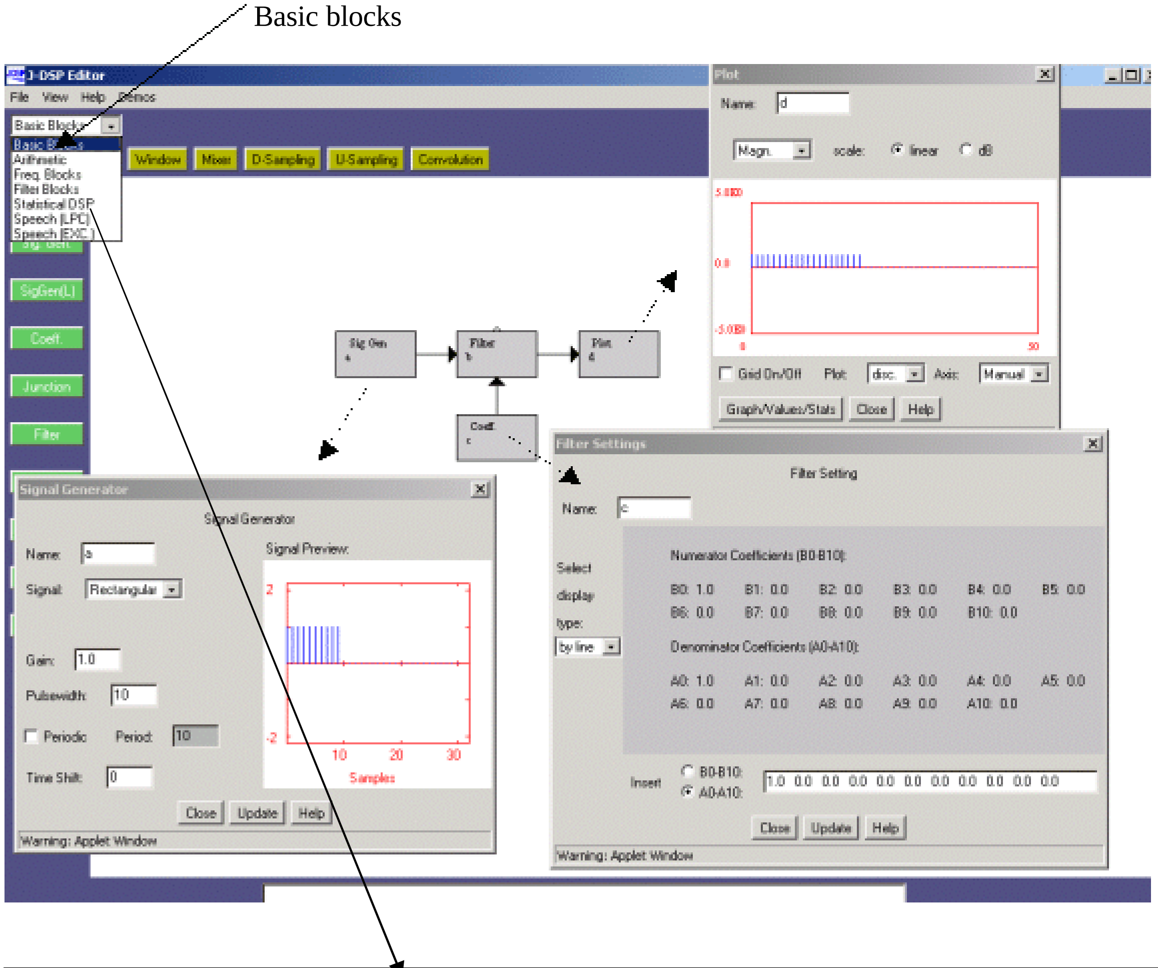


Fig. 3. J-DSP buttons for a source-filter simulation.

Note that each block has signal input(s), designated by the small triangular nodes, on the left and signal output(s) to the right. Some blocks carry parameter inputs and outputs at the bottom and top of the block respectively. For example, the ***Filter*** block has a coefficient input on the bottom and a coefficient output on the top. To select a block, click once to highlight it. You can then move it by placing the mouse arrow over it, holding down the left mouse button and dragging the box to a new location. To delete a block, simply select it and press the "delete" key on your keyboard. To link blocks, click once inside the small triangle on the right side of the signal generator box and while holding the mouse button down, drag the mouse arrow to the triangle on the left side of the filter box. Release the mouse button to create a connection between the two boxes. Always make the connections in the direction of the signal flow. Now connect the ***Coeff***. block that is used to specify filter coefficients, to the ***Filter*** block's parameter input as shown in Fig. 4. Next, connect the ***Filter*** block to the ***Plot*** block so that your editor window looks like the block diagram in Fig. 4. Note that you can view the dialog box of each block by double-clicking on the block, as shown in Fig. 4.


Basic blocks

Plot — Name: d; Magn.; scale: ● linear ○ dB; □ Grid On/Off; Plot: disc.; Axis: Manual

Signal Generator — Name: a; Signal: Rectangular; Gain: 1.0; Pulsewidth: 10; □ Periodic; Period: 10; Time Shift: 0

Filter Setting — Name: c; Select display type: by line

Numerator Coefficients (B0-B10):
B0: 1.0 B1: 0.0 B2: 0.0 B3: 0.0 B4: 0.0 B5: 0.0
B6: 0.0 B7: 0.0 B8: 0.0 B9: 0.0 B10: 0.0

Denominator Coefficients (A0-A10):
A0: 1.0 A1: 0.0 A2: 0.0 A3: 0.0 A4: 0.0 A5: 0.0
A6: 0.0 A7: 0.0 A8: 0.0 A9: 0.0 A10: 0.0

Insert ○ B0-B10: ● A0-A10: 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

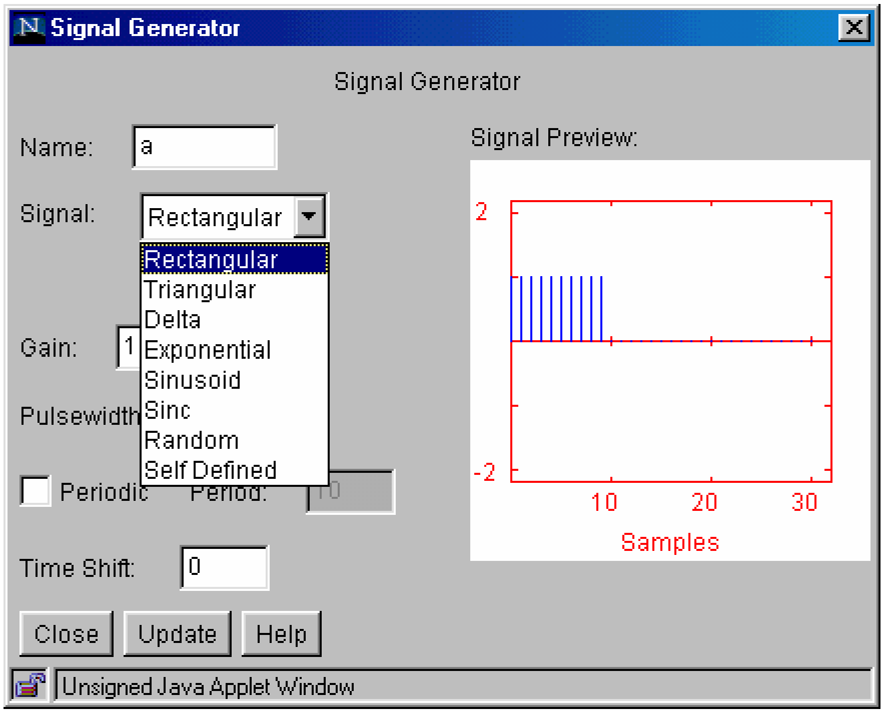
Please note that the following notation has been used throughout this document:
Block names: bold and italic, e.g., ***Plot***
Drop down menu item name: large bold font, e.g., **Basic Blocks**
Button: third brackets, e.g., [update]
Option to be chosen by user in a dialog box of a block: inverted comma, e.g., "Gain"

Fig. 4. Dialog windows in J-DSP.

1.3. Example J-DSP laboratory assignment.

This assignment assumes some familiarity with basic DSP concepts. It continues from section 1.2 to build on the block diagram created there.

Start with the block diagram of Fig. 4. Let us now form a signal using the signal generator. Double click inside the ***SigGen*** box and a dialog window will appear. This window is shown in Fig. 5. If you do not see a dialog window, you are using an older Internet browser and must download the newest version of Netscape or Internet Explorer and start over. Use Internet Explorer 5.5 or later, or Navigator version 4.6 or later, with its Java plug in.



Name: a

Signal: Rectangular

Gain: 1

Pulsewidth

□ Periodic Period: 10

Time Shift: 0

Fig. 5. Signal generator dialog

On the right side of the signal generator window, you can see a preview of the signal. You may change the “name” of the signal, the “gain”, the “pulse width”, the “period” and the “time shift” by typing the desired value into the appropriate box. The signal type can be changed by clicking on the drop-down menu and selecting a signal. If you select a User-defined signal, an [Edit signal] button will appear allowing you to edit the signal. With all signals except audio, J-DSP assumes a normalized sampling frequency of 1Hz. Hence the sampling frequency in terms of radians is 2π. All frequencies are entered as a function of π, e.g., 0.1π, 0.356π, etc. Any sinusoidal frequency at or above π will result in aliasing.

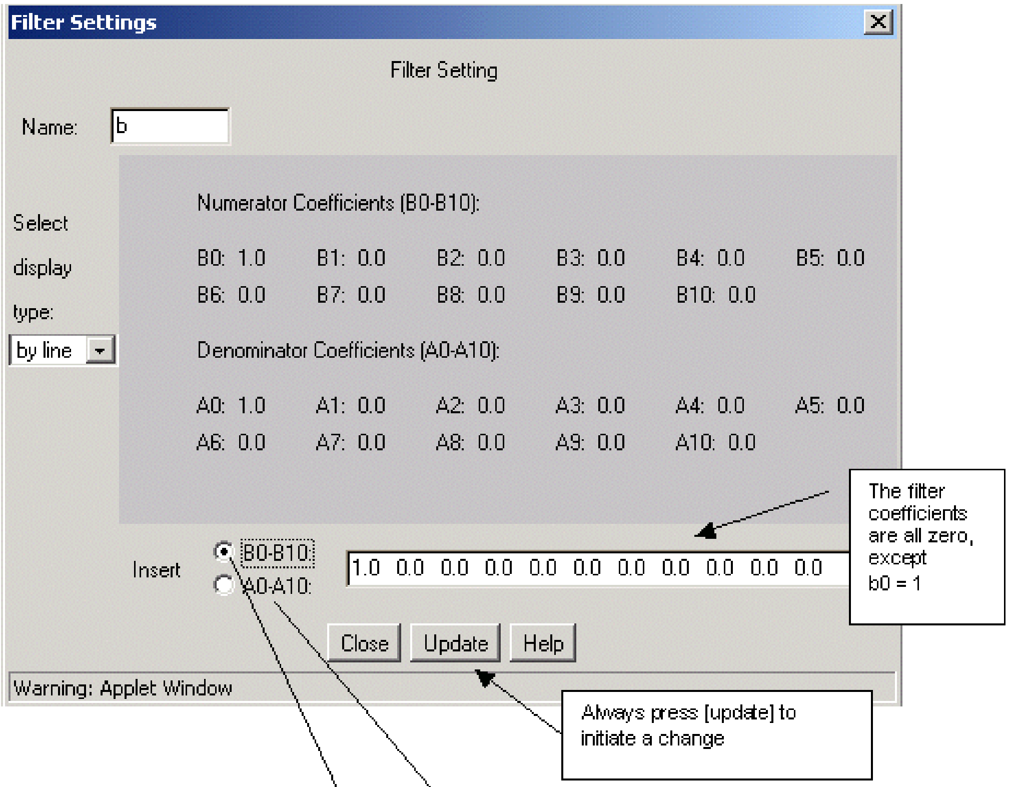
Step 1.1: Create a sinusoid with “frequency” 0.1π, “amplitude” 3.75, “pulse width” 40. When all of the parameters have been entered, press the [Update] button to update the signal preview. Remember that whenever changes are made to this box, the [Update] button must be pressed in order for the changes to take effect. On the right, you can see a preview of the input signal. Count the number of samples within a period. How many do you have? (ans: 20 samples).

Step 1.2: Create a sinusoid with “frequency” π, “amplitude” 3.75, “pulse width” 40 (remember to press update for changes to take place). What happens? (ans: we have aliasing, i.e, no signal).

Step 1.3: Create a sinusoid with “frequency” 1.3π, “amplitude” 3.75, “pulse width” 40. What happens? Count the number of samples in a period. (ans: we have aliasing again , signal makes no sense).

Step 2: Next, we want to take a look at the ***Filter*** output in the time and frequency domain. Set the values in ***SigGen*** as per step 1.1. Double-click the ***Plot*** block and a new dialog window will appear. You should again see the input signal because the filter is just letting the signal pass through unaffected, since no coefficients have been set. If you press the [Graphs/Values/Stats] button, a table with the values of the signal appear. In the first column you see the indices of the samples and the second column shows you the values. Close the value dialog box.

Step 2.1: Let us now see the filter in action. Keep the ***Plot*** window open to observe any changes. Double click the ***Coeff***. block. You should see the dialog window of Fig. 6.



Filter Settings

Filter Setting

Name: b

Select display type: by line

Numerator Coefficients (B0-B10):

B0: 1.0	B1: 0.0	B2: 0.0	B3: 0.0	B4: 0.0	B5: 0.0
B6: 0.0	B7: 0.0	B8: 0.0	B9: 0.0	B10: 0.0	

Denominator Coefficients (A0-A10):

A0: 1.0	A1: 0.0	A2: 0.0	A3: 0.0	A4: 0.0	A5: 0.0
A6: 0.0	A7: 0.0	A8: 0.0	A9: 0.0	A10: 0.0	

Insert ● B0-B10: ○ A0-A10: 1.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0 0.0

Warning: Applet Window

The filter coefficients are all zero, except b0 = 1

Always press [update] to initiate a change

Note that the filter coefficients correspond to the following difference equation.

$$y(n) = \sum_{i=0}^{L} b_i x(n-i) - \sum_{i=0}^{M} a_i y(n-i)$$

Fig. 6. Coefficient entry in J-DSP

Step 2.2: Keep the values in ***SigGen*** as per step 1.1. Change the filter coefficient to b0=4 and press [Update]. Double click on the ***Plot*** block. You should see that the amplitude of the sinusoid has changed (ans: peak amplitude 4x3.75= 15).

Step 2.3: Implement a pure delay by setting b5=1 and rest of the coefficients (including b0) to zero and press [Update]. What happens to the sinusoid?

Step 2.4: Implement a simple LPF, set b0 = 0.2 and a1 = -0.8 and press [Update]. Generate a sinusoid with “gain” 1, “frequency” 0.1π, “pulse width” 256. What do you observe? What kind of signal do you get at the output? Why? What is the peak-to-peak value? Do we have a change? Is there a phase shift? What filter function determines the time shift?

Step 2.5: Select the ***Freq-Resp*** block from the panel of general blocks on the left of the window and place it to the north of the ***Filter*** block. Connect the parameter output to the ***Freq-Resp*** block. Double click the ***Freq-Resp*** block. You should see the magnitude and phase response of the filter. Change the coefficient to a1 = 0.8 instead of a1 = -0.8. What do you see in the frequency response and output? (ans: HPF, decrease in amplitude)

Step 3: To view the signal in the frequency domain, insert an ***FFT*** block between the ***Filter*** and the ***Plot*** boxes as shown below in Fig. 7. The ***FFT*** block can be found under the **Freq. Blocks** menu.

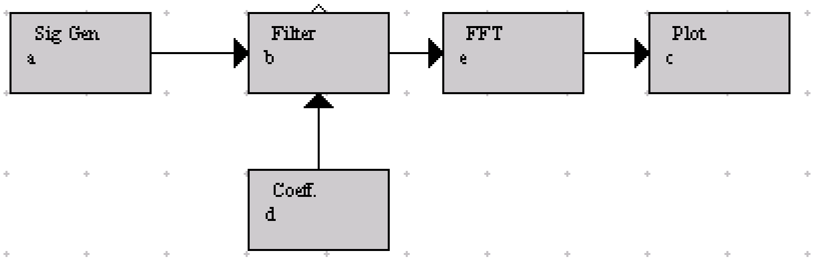


Fig. 7. Source-filter simulation with FFT at the output

Step 3.1: Set the ***Filter*** parameters and input as per step 2.4. Double click on the ***FFT*** block and change the “FFT size” to 256 points and then press [Close]. Now, you can see the magnitude and the phase of the signal in the frequency domain. The magnitude has a sharp peak approximately at 0.31, the frequency of our sinusoidal signal (0.1x3.1459).

Step 3.2: Change the sinusoidal frequencies as per steps 1.2 and 1.3 but with “pulse width” 256. What do you observe?

Step 3.3: Delete the Filter block. Set the sinusoidal “frequency” in ***SigGen*** as per step 1.1 but with “pulse width” 256. Now create a second ***SigGen*** block and a ***Mixer*** block. Your editor window should contain a block diagram that looks like the one in Fig. 8.

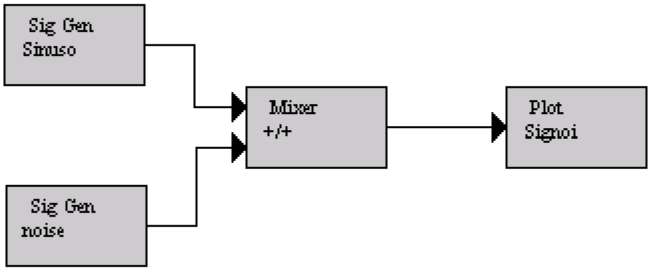
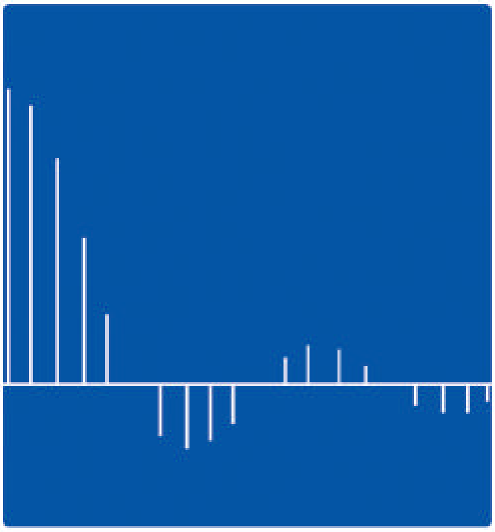


Fig. 8. Sinewave plus noise simulation

Change the name of the first ***SigGen*** block to 'Sinusoi', the second ***SigGen*** block to 'noise' and the **Plot** block to 'SigNoi'. The names are restricted to six characters. Following that, we edit the ***SigGen*** block called 'noise'. Open the dialog window and change the "signal type" to "random". Choose a "variance" of 4 and extend the "pulse width" to 256 samples, in order to have noise over the full length of the signal. Now take a look at the output signal. In the time domain it is very hard to see that a sinusoid is present. However, if you view the signal in the frequency domain with an FFT size of 256, then you still find a peak at approximately 0.31.

Step 3.4: Change the amplitude of the sinusoid up or down and observe the spectra (FFT plot). Try different values to make the sinusoid to dominate the noise signal or be masked by the noise signal. Remember the movie "The Hunt for Red October" which was about a stealth Soviet submarine defecting? In that movie they showed sonar operators viewing FFT spectra and listening to sonar signals as they were searching for submarine propeller signatures (quasi-periodic signals) in ocean noise (random broadband signals). Stealth submarines have, among other things, weak broadband propeller signatures that can be masked easily by ocean noise.



J-DSP® Editor

MANUAL

ARIZONA STATE UNIVERSITY

J-DSP Editor concept by: Prof. A. Spanias

Copyright © 1997-2003: Arizona Board of Regents

Andreas Spanias

Biography

Andreas Spanias is Professor in the Department of Electrical Engineering at Arizona State University (ASU). His research interests are in the areas of adaptive signal processing and speech processing. While at ASU, he has developed and taught courses in DSP, adaptive signal processing, and speech coding. He has also developed and taught continuing education short courses in digital signal processing and speech coding. He has developed a software concept called Java-DSP that is intended for use in undergraduate and graduate education (http://jdsp.asu.edu). Andreas Spanias has been the principal investigator on research contracts from Intel Corporation, Sandia National Labs, Motorola Inc., and Active Noise and Vibration Technologies. He has also consulted with Inter-Tel Communications, Intel Corporation, Motorola, Texas Instruments, and the Cyprus Institute of Neurology and Genetics. He is member of the DSP Committee of the IEEE Circuits and Systems society, and has served as a member in the technical committee on Statistical Signal and Array Processing of the IEEE Signal Processing society. He has also served as Associate Editor of the IEEE Transactions on Signal Processing and as General Co-chair of the 1999 International Conference on Acoustics Speech and Signal Processing (ICASSP-99) in Phoenix. He served as the IEEE Signal Processing Vice-President for Conferences and the Chair of the Conference Board. He served as a member of the IEEE Signal Processing Executive Committee and as Associate Editor of the IEEE Signal Processing Letters. He is currently serving as a member of the IEEE SPS Publications Board and as a member-at-large of the IEEE SPS Conference Board. He has been Chair of the IEEE Communications and Signal Processing Chapter in Phoenix, and is a member of Eta Kappa Nu, and Sigma Xi. Andreas Spanias is recipient of the 2002 IEEE Donald G. Fink paper prize award. He also received the 2003 teaching award from the Phoenix chapter for his contributions in on-line laboratories using J-DSP and has been recipient of various other research awards from Intel Corporation. He is Distinguished Lecturer of the IEEE SPS for 2004 and was recently elected Fellow of the IEEE.

Research Contributions Explained

Andreas Spanias has contributed to DSP and particularly to speech processing and related industry applications. He developed a unique mixed-basis signal analysis-synthesis method [17] along with a novel algorithm that selects iteratively the constituent narrowband and broadband basis functions. He applied this method to low-rate speech coding in [16] and demonstrated improved speech quality relative to other transform coders. In another contribution, he and his students improved significantly upon the well known sinusoidal speech analysis-synthesis model by using non-minimum phase modeling techniques [9] along with improved pitch estimation algorithms [7]. This work resulted in speech coding implementations that are reported in [5]. He also made important contributions to speech enhancement and speech recognition. He and his students developed a novel speech enhancement method that uses a harmonic sinusoidal model whose parameters are estimated using a hidden Markov model (HMM) [10]. This perceptually motivated noise reduction method improves significantly upon the state of the art by attenuating the noise components between the harmonics and by exploiting the noise masking effects of the sinusoids. He also developed s new segmentation technique [12] and other auxiliary algorithms that improve alphabet recognition [6,8,11]. Andreas Spanias published comprehensive survey papers on audio coding [1] and speech coding [2] in the *Proceedings of the IEEE*. As evidence of the quality of his work it is noted that publication [1] received the *prestigious 2002 Donald Fink IEEE-wide paper prize award*. Andreas Spanias collaborates extensively with industry. He has established and directed a five year 1.5-million dollar ASU program funded by Intel Corp. towards the design of the low-power DSP core 60172® and its multiprocessor version named Phoenix® architecture. His work involved chip design and analysis. He made detailed design recommendations [3] on the arithmetic unit, instruction set, architecture interface and on-chip memory to accommodate low-power implementation of wireless telecom standards involving source and channel coding as well as other physical layer functions. He also provided a detailed analysis on the numerical behavior of the chip and developed algorithms to handle numerical problems in the fixed-point realization of the source and channel coders**.**

The two papers of Andreas Spanias in the Proc. of the *IEEE* [1,2] provide a comprehensive survey of state-of-the-art algorithms and research in speech and audio coding. Paper [1] received a paper prize award from the IEEE board of directors. Andreas Spanias has developed an extensive suite of MATLAB speech coding software tools for public-domain use that have been posted on the web and downloaded by more than 5000 students, scientists, faculty, and engineers in the speech processing community. The industry collaboration of Andreas Spanias on chip design [3] has had a lasting impact. The unique multi-processor architecture, Intel Phoenix®, that was designed in part by Andreas Spanias and his ASU team, includes two 60172® DSP cores and a micro-controller along with a hardware Viterbi accelerator. An enhanced version of this architecture is currently being promoted for third and fourth generation wireless cellular systems. As part of this work, Andreas Spanias and his team have also developed a series of modified linear prediction analysis-by-synthesis algorithms accompanied by a large body of fixed-point software for coding applications. An algorithm and portions of this software found their way to the Intel ProShare**®** videoconferencing product. As evidence of the great impact of his work for Intel, Andreas Spanias received a series of awards from Intel Corporation citing technical leadership and outstanding contributions. *(® Registered by Intel Corp.)*

Other individual contributions are:

- Significant contributions in adaptive signal processing; time-varying spectral estimation [15], adaptive antenna arrays [4], frequency-domain adaptive filtering [14,18], system identification based on ellipsoidal bounded error constraints [13].
- He developed a new concept for on-line DSP laboratories using a novel Java object-oriented programming environment called Java-DSP. This concept was adopted by ten university and industry partners and was recently awarded a sizeable three-year NSF grant for disseminating and extending J-DSP to communications and other areas.
- He demonstrated outstanding and sustained leadership in academic, industry, and IEEE activities as evidenced by various project, chip development, and IEEE paper awards as well as devoted service in upper level IEEE volunteer activities.

Select Papers Published

[1] T. Painter and A. S. Spanias, “Perceptual coding of digital audio,” *Proc. IEEE*, vol. 88, no.4 , pp. 451-513, Apr. 2000. This 63-page *award-wining paper* was the main article featured in the April 2000 issue of the *Proc. of the IEEE*. It describes the theory and research on MPEG audio and cinema algorithms and has more than 400 references. This paper won the prestigious **2002 IEEE Donald G. Fink Prize Paper Award** for best survey paper across all IEEE societies and publications. *(A. Spanias principal investigator and Ph.D. advisor)*

[2] A. S. Spanias, “Speech coding: a tutorial review, ” *Proc. of the IEEE*, vol. 82, no. 10, pp. 1441-1582, Oct. 1994. *(A. Spanias principal investigator)*

[3] M. Deisher, P. Loizou, G. Lim, A. S. Spanias and B. Mears, “A new highly integrated architecture for speech processing and communication applications, ” *Intel Technical Journal*, Special Issue on Computer Supported Cooperation, pp. 41-56, Spring 1994.

[4] S. Bellofiore, J. Foutz+, C. Balanis, A. S. Spanias, J. Capone, and T. Duman, "Smart Antenna System Analysis, Integration, and Performance for Mobile Ad-Hoc Networks (MANETs)," Full paper to appear in *IEEE Trans. Antennas Propagation, Special issue on Wireless Communications*, vol. 50, no. 3, March 2002 (results of 2-year NSF-ITR project; A. Spanias Co-PI)

[5] S. Ahmadi+ and A. Spanias, “Algorithms for Low-bit rate sinusoidal coding,” *Speech Communications, Vol. 34(2001), pp.369-390, June 2001 (A. Spanias principal investigator and Ph.D. advisor)*

[6] Min-Tau Lin^{+}, A. S. Spanias, and P. Loizou^{+}, “Speech Recognition Using Minimum Error Classification," *Speech Communication*, vol. 30, pp. 27-36, January 2000 (A. Spanias principal investigator and Ph.D. advisor)

[7] S. Ahmadi^{+} and A.S. Spanias, "Cepstrum-Based Pitch Detection Using a New Statisctical V/UV Classification Algorithm**,"** *IEEE Trans. on Speech and Audio*, Vol. 7, No. 3, pp. 333-338, May 1999 *(A. Spanias principal investigator and Ph.D. advisor)*

[8] P. Loizou+ and A. S. Spanias, “Improved speech recognition using a subspace projection approach,” *IEEE Trans. on Speech and Audio Processing*, vol. 7, no. 3, pp. 343-345, May 1999. *(A. Spanias principal investigator and Ph.D. advisor)*

[9] S. Ahmadi+ and A. S. Spanias, “A new phase model for sinusoidal transform coding,” *IEEE Trans. on Speech and Audio Processing*, vol. 6, no. 5, pp. 495-501, Sept. 1998. *(A. Spanias principal investigator and Ph.D. advisor)*

[10] M. Deisher+ and A. S. Spanias, “Speech enhancement using state-based estimation and sinusoidal modeling,” *Journal of Acoustical Society of America*, vol. 102(2), pp. 1141-1148, Aug. 1997. *(A. Spanias principal investigator and Ph.D. advisor)*

[11] P. Loizou+ and A. Spanias, "Improving discrimination of confusable words using the divergence measure,” *Journal of Acoustical Society of America*, Vol, 101(2), pp. 1106-1111, Feb. 1997. *(A. Spanias principal investigator and Ph.D. advisor)*

[12] P. Loizou+ and A. S. Spanias, “High Performance Alphabet Recognition,” *IEEE Trans. on Speech and Audio Processing*, vol. 4, no. 6, pp. 430-445, Nov. 1996. *(A. Spanias principal investigator and Ph.D. advisor)*

[13] K. Tsakalis, M. Deisher+, and A. S. Spanias, “System identification based on bounded error constraints,” *IEEE Trans. on Signal Processing*, vol. 43, no. 12, pp. 3071-3075, Dec. 1995. *(A. Spanias co-principal investigator and Ph.D. advisor)*

[14] M. Deisher+ and A. S. Spanias, “Practical considerations in the implementation frequency-domain adaptive noise cancellation,” *IEEE Trans. on Circuits and Systems, Part II: Analog and Digital Signal Processing*, vol. 41, no. 2, pp. 164-168, Feb. 1994. *(A. Spanias principal investigator and Ph.D. advisor)*

[15] A. S. Spanias, “A block time and frequency modified covariance algorithms for spectral analysis,” *IEEE Trans. on Signal Processing*, vol. 41, no. 11, pp. 3138-3153, Nov. 1993. *(A. Spanias principal investigator)*

[16] A. S. Spanias and P .Loizou+, “A mixed Fourier/Walsh transform scheme for speech coding at 4 kbps,” *Proc. IEE-Part I (Communications, Speech, and Vision)*, vol. 139(5), pp. 473-481, Oct. 1992. *(A. Spanias principal investigator and Ph.D. advisor)*

[17] A.S. Spanias, "A Hybrid Transform Method for Speech Analysis and Synthesis," *Signal Processing*, Vol. 24, pp. 217-229, Aug. 1991 *(A. Spanias principal investigator)*

[18] W. Mikhael and A. S. Spanias, "A fast frequency-domain adaptive algorithm," *Proc. of the IEEE*, vol. 76, no. 1, pp. 80-82, Jan. 1988.

Complete Curriculum Vita

Andreas Savva Spanias, Professor
Department of Electrical Engineering

Background

Personal

U.S. Citizen, Married to Photini Spanias, Ed.D., Two Children: John (13) and Louis (10)

Education

- Ph.D. 1988, Dept. of Electrical and Computer Eng., West Virginia University,
- M.S.E.E. 1985, Dept. of Electrical and Computer Eng., West Virginia University
- B.S.E.E. 1983, Dept. of Electrical and Computer Eng., West Virginia University
- HTI Diploma, 1979, Diploma in Electrical Engineering, Higher Technical Institute (HTI), Nicosia, Cyprus.

Areas of Teaching and Research

- Teaching: Digital Signal Processing, Linear Systems, Speech Processing, Adaptive Signal Processing, Spectral Estimation, Communications.

- Research: Adaptive Filters, Speech Analysis/Synthesis, Coding, and Enhancement, Voice Processing Algorithms for Multimedia Applications, Time-Varying Spectral Analysis, DSP Architectures.

Positions Held

- Aug. 1997-present Professor, Department of Electrical Engineering, Arizona State University.
- Aug. 1993-July 1997 Associate Professor, Department of Electrical Engineering, Arizona State University.
- Aug. 1988-July 1993 Assistant Professor, Department of Electrical Engineering, Arizona State University.
- May 1985-Aug. 1988 Graduate Research Assistant - Dept. of Electrical and Computer Engineering, WVU.
- Aug. 1983-May 1985 Graduate Teaching Assistant - Dept. of Electrical and Computer Engineering, WVU.
- July 1979-Aug. 1981 Tactical Communications Engineer - National Guard.

Honors, Awards, Memberships

- 2003 IEEE Fellow
- 2004 IEEE Distinguished Lecturer in signal processing
- 2002 IEEE Donald G. Fink Prize Paper Award from the IEEE Board of Directors for the paper “Perceptual Coding of Digital Audio”
- 2003 Teaching Award, "for contributions to J-DSP," IEEE Phoenix Chapter, Phoenix - January 2003
- 2002 Researcher of the Year Award, IEEE Phoenix Chapter, Phoenix - January 2002
- 1997 Award from the Intel Advanced Personal Communications Division - Central Logic Engineering, "Team Recognition Award for outstanding support and leadership of the ASU Team in the Intel GSM Cellular Mobile Telephone Project"
- 1996 Award from Intel Corporation (Portland) “In Appreciation of Support for the Intel Research Program”
- 1993 Award from Intel Corporation for "Leadership and Contributions in the Development of the Intel 60172 Signal Processing Architecture"
- 1992 Team Award from Intel Corporation to “Andreas Spanias and three of his graduate students for contributions in the development of speech processing algorithms.”
- 1983 Rufus West Achievement Award at WVU

- Member Sigma Xi, Tau Beta Pi, HKN, Golden Key.

Publications

Refereed Archival Journal Papers

a. **Published or Accepted for Publication**

J.1. Ted Painter^{+} and A. Spanias, " Sinusoidal Analysis-Synthesis of Audio using Perceptual Criteria," To appear in the *IEEE Transactions on Speech and Audio*

J.2. Gopal Nair^{+} and A. Spanias,"The Eigenspace Projection Algorithm," *Signal Processing*, Accepted and will appear in 2003

J.3. J. Foutz, A. Spanias, S. Bellofiore and C Balanis, " Adaptive Eigen-Projection Beamforming Algorithms for 1-D and 2-D Antenna Arrays, Accepted in IEEE Antennas and Propagation Letters, 2003

J.4. Ted Painter and Andreas Spanias, "Sinusoidal Analysis-Synthesis of Audio Using Perceptual Criteria," EURASIP JASP - Special Issue On Multimedia Signal Processing, Vol. 2003, Issue 1, pp. 15-20, January 2003

J.5. S. Bellofiore, J. Foutz, C. Balanis, A,S, Spanias, T, Duman, J, Capone "Smart Antennas for Mobile adhoc Networks, IEEE Trans. on Antennas and Propagation, pp. 571-581, Vol. 50, No. 5, May 2002

J.6. S. Bellofiore, C. Balanis, J. Foutz, and A,S, Spanias, "Smart Antennas Systems for Mobile Communications Networks: Part 1: Overview of the Antenna Design," IEEE Antennas and Propagation Magazine, pp. 145-154, Vol. 44, No. 3, June 2002

J.7. A. Kitsios A. Spanias, and B. Welfert, "Fast modified covariance algorithm with individual step sizes," Signal Processing 82(5), pp. 715-7120, June 2002

J.8. S. Bellofiore, C. Balanis, J. Foutz, and A,S, Spanias, "Smart Antennas Systems for Mobile Communications Networks: Part 2: Algorithms," IEEE Antennas and Propagation Magazine, pp. 106 -114, Vol. 44, No. 4, August 2002

J.9. A. Kitsios^{+} and A. Spanias, "Fast modified covariance algorithm with individual step sizes," *Signal Processing*, Accepted and will appear in 2002

J.10. S. Ahmadi^{+} and A. Spanias, "Algorithms for Low-bit rate sinusoidal coding," *Speech Communications, 34(2001), pp. 369-390, June 2001*

J.11. Ted Painter^{++} and Andreas S. Spanias, "Perceptual Coding of Digital Audio," *Proceedings of the IEEE*, pp. 451-513, Vol. 88, No.4, April 2000 (**winner of 2002 IEEE Donald G. Fink Prize Paper Award**)

J.12. A. Spanias, S. Urban, A. Constantinou^{+}, M. Tampi^{+}, X. Zhang^{+}, M. Tampi^{+}, C. Stilianou^{+}, "Development of a Web-based Signal and Speech Processing Laboratory for Distance Learning," *ASEE Computers in Education Journal*, pp. 21-26, Vol. X, No.2, April-June 2000

J.13. Min-Tau Lin^{+}, A. S. Spanias, and P. Loizou^{+}, "Speech Recognition Using Minimum Error Classification," *Speech Communication*, vol. 30, pp. 27-36, January 2000

J.14. S. Ahmadi^{+} and A.S. Spanias, "Cepstrum-Based Pitch Detection Using a New Statisctical V/UV Classification Algorithm**,"** *IEEE Trans. on Speech and Audio*, Vol. 7, No. 3, pp. 333-338, May 1999

J.15. P. Loizou^{+} and A.S. Spanias, "Improved speech recognition using a subspace projection approach*," IEEE Trans. on Speech and Audio*, vol. 7, no. 3, pp. 343-345, May 1999

J.16. S. Ahmadi^{+} and A.S. Spanias, "A New Phase Model for Sinusoidal Transform Coding of Speech," *IEEE Trans. on Speech and Audio*, vol. 6, no. 5, pp. 495-501, Sept. 1998

J.17. M. Deisher^{+} and A.S. Spanias, "Speech Enhancement using state-bases estimation and sinusoidal modeling," *Journal of Acoustical Society of America*, vol. 102(2), pp. 1141-1148, Aug. 1997

J.18. P. Loizou[1] and A. Spanias, "High Performance Alphabet Recognition," *IEEE Trans. on Speech and Audio,* vol. 4, no. 6, pp. 439-445, Nov. 1996 ,

J.19. P. Loizou[+] and A. Spanias, "Improving discrimination of confusable words using the divergence measure," *Journal of Acoustical Society of America*, Vol, 101(2), pp. 1106-1111, Feb. 1997

J.20. Q. Shen[+] and A. Spanias, "Adaptive Active Sound Reduction, *Noise Control Engineering Journal*, J44 (6), pp. 281-293, Nov. 1996

J.21. A. Spanias and T. Painter, "An Educational Software Tool for the Study of Speech Coding Algorithms in a DSP Class," Special Issue on DSP Education, *IEEE Trans. on Education*, Vol. 39, pp. 143-152, May 1996

J.22. K. Tsakalis, M. Deisher[+], and A. Spanias, "System Identification Based on Bounded Error Constraints, *IEEE Transactions on Signal Processing*, Vol. 43, No. 12, pp. 3071-3075, Dec. 1995

J.23. J. Liu[+], A. Spanias, and J. Maisel,"A Real-time Adaptive Interference Canceller using the BLMS Algorithm," *American Society of Engineering Education (ASEE) J. Eng. Tech.*, Vol. 12, No. 1, pp. 34-38, Spring 1995.

J.24. P. Loizou[+], M. Dorman, and A. Spanias, "Automatic recognition of syllable-final nasals preceded by e," *Journal of Acoustical Society of America*, Vol. 97(3), pp. 1925-1928, March 1995.

J.25. A.S. Spanias, "Speech Coding: A Tutorial Review," *Proceedings of the IEEE,* Vol. 82, No. 10, pp. 1441-1582, October 1994.

J.26. M. Deisher[+] and A.S. Spanias, "Practical Considerations in the Implementation Frequency-Domain Adaptive Noise Cancellation," *IEEE Transactions on Circuits and Systems*, Part II: Analog and Digital Signal Processing, Vol. 41, No. 2, pp. 164-168, Feb. 1994.

J.27. A. Spanias, M. Deisher[+], P. Loizou[+], G. Lim[+], and B. Mears, "A New Highly Integrated Architecture for Speech Processing and Communication Applications," *Intel Technology Journal - Special Issue on Computer Supported Cooperation,* pp. 41-56, Spring 1994.

J.28. A.S. Spanias, *IEEE Transactions on Signal Processing*, "A Block Time and Frequency Modified Covariance Algorithms for Spectral Analysis," vol. 41, No. 11, pp. 3138-3153, Nov. 1993.

J.29. A. Spanias, P. Loizou[+], G. Lim[+], Y. Chen[+], G. Hu[+], "Analysis/Synthesis of Speech using the Short-Time Fourier Transform and a Time-varying ARMA process," *Trans. of the Institute of Electronics, Information and Communication Engineers (IEICE),* Vol. E76-A, No. 4, pp. 645-652, April 1993

J.30. A.S. Spanias and *P.C. Loizou*, "A Mixed Fourier/Walsh Transform Scheme for Speech Coding at 4 KBPS," *Proc. IEE-Part I (Communications, Speech, and Vision),* Vol. 139(5), pp. 473-481, Oct. 1992.

J.31. A.S. Spanias, "A Frequency Selective Adaptive Algorithm," *Journal of Computers EE - Special Issue on Adaptive Signal Processing,* Editors: D. Etter and M. Ahmadi, pp. 301-313, Vol. 18, No. 3/4, May/July 1992.

J.32. A.S. Spanias, S.B. Jonsson[+] and S.D. Stearns, "Transform Methods for Seismic Data Compression," *IEEE Trans. on Geoscience and Remote Sensing,* Vol. GARS-29, No. 3, pp. 407-417, May 1991

J.33. A.S. Spanias and F.H. Wu, "Speech Coding and Recognition: A Review," *Trans. IEICE - Special Issue on Fundamentals of Next Generation Human Interface*, Ed: Sadaoki Furui, pp. 132-148, Feb. 1992

[+] Names of student co-authors that have been supervised by A. Spanias

J.34. A.S. Spanias, "A Hybrid Transform Method for Speech Analysis and Synthesis," *Signal Processing*, Vol. 24, pp. 217-229, Aug. 1991.

J.35. W.B. Mikhael and A.S. Spanias, "Accurate Representation of Time-Varying Signals using Mixed Transforms with Applications to Speech," *IEEE Trans.on Circuits and Systems*, Vol. 36, No. 2, pp. 329-331, Feb. 1989.

J.36. W.B. Mikhael and A.S. Spanias, "Efficient Modeling of Dominant Transform Components Representing Time-Varying Signals," *IEEE Trans. on Circuits and Systems*, Vol. 36, No. 2, pp. 331-334, Feb. 1989.

J.37. W.B. Mikhael and A.S. Spanias, "A Fast Frequency-Domain Adaptive Algorithm," *Proc. of the IEEE*, Vol. 76, No. 1, pp. 80-82, Jan. 1988.

J.38. W.B. Mikhael and A.S. Spanias, "Comparison of Several Frequency-Domain LMS Algorithms," *IEEE Trans. on Circuits and Systems*, Vol. 34, No. 5, pp. 586-588, May 1987.

J.39. W.B. Mikhael, A.S. Spanias, G. Kang and L. Fransen,"A Two-Stage Pole-Zero Predictor," *IEEE Trans. on Circuits and Systems*, Vol. 33, No. 3, pp. 352-354, March 1986.

J.40. S. Miller and A.S. Spanias, "Antenna Beamforming using Adaptive Quiescent Pattern Control," Submitted to *EURASIP Journal on Applied Signal Processing (EURASIP JASP* ***- Special Issue on Smart Antennas), Submitted May 2003***

J.41. S. Miller and A.S. Spanias, " N. Chakravarti, A. Spanias, ,L.D. Iasemidis, and K. Tsakalis, “AR Modeling of DNA sequences," Submitted to *EURASIP Journal on Applied Signal Processing (EURASIP JASP* ***- Special Issue on Genomic Signal Processing), Submitted Feb 2003***

Refereed Papers in National and International Conference Proceedings

C.1 B. Badke and A. Spanias, Partial Band Interference Excision For GPS Using Frequency-Domain Exponents, *IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP-2002),* Vol 4 , pp. 3936 –3939, Orlando, May 2002.

C.2 N. Chakravarti, A. Spanias, ,L.D. Iasemidis, and K. Tsakalis, “AR Modeling of DNA sequences,” 22nd IASTED International Conference MIC 2003, 538-542, Innsbruck, Austria February 10-13, 2003

C.3 T. Thrasyvoulou, K. Tsakalis, and A. Spanias, “J-DSP-Control: A Control Systems Simulation Environment,” 22nd IASTED International Conference MIC 2003, 538-542, Innsbruck, Austria February 10-13, 2003

C.4 Balaji Veeramani, K. Narayanan, Awadhesh Prasad, A. Spanias and L. D. Iasemidis, "On the use of the directed transfer function for nonlinear systems", Proceedings of IASTED (International Association of Science and Technology for Development) International Conference, Palm Springs, California, USA, Feb. 24-26, 2003, pp. 270-274.

C.5 Balaji Veeramani, Awadhesh Prasad, K. Narayanan, A. Spanias and L. D. Iasemidis, “Measuring information flow in nonlinear systems - A modeling approach in the state space”, Proceedings of the 40th Annual Rocky Mountain Bioengineering Symposium, Biloxi, Mississipi, USA, April 11-13, 2003, in press.

C.6 Rajeshkumar Venugopal, K. Narayanan, A. Prasad, A. Spanias, J.C. Sackellares and L.D. Iasemidis, “A new approach towards predictability of epileptic seizures : KLT dimension", Proceedings of the 40th Annual Rocky Mountain Bioengineering Symposium, Biloxi, Mississipi, USA, April 11-13, 2003, in press.

C.7 Rajeshkumar Venugopal, A. Prasad, K. Narayanan, A. Spanias, & L.D. Iasemidis, “Nonlinear noise reduction and predictability of epileptic seizures”, Proceedings of IASTED (International Association of Science and Technology for Development) International Conference, Palm Springs, California, USA, Feb. 24-26, 2003, pp. 240-245.

C.8 Shivkumar Sabesan, K. Narayanan, Awadhesh Prasad, A. Spanias and L. D. Iasemidis, “Improved measure of information flow in coupled nonlinear systems”, Proceedings of IASTED (International Association of Science and Technology for Development) International Conference, Palm Springs, California, USA, Feb. 24-26, 2003, pp. 329-333.

C.9 Shivkumar Sabesan, K. Narayanan, Awadhesh Prasad, A. Spanias, J.C. Sackellares & L. D. Iasemidis, “Predictability of epileptic seizures: A comparative study using Lyapunov exponent and entropy based measures”, Proceedings of the 40th Annual Rocky Mountain Bioengineering Symposium, Biloxi, Mississipi, USA, April 11-13, 2003, in press..

C.10 T. Thrasyvoulou, K. Tsakalis and A. Spanias, “J-DSP-C, A Control Systems Simulation Environment For Distance Learning: Labs And Assessment,” 33rd ASEE/IEEE Frontiers in Education Conference, Boulder, CO, November 5-8, 2003.

C.11 V. Atti and A. Spanias, “A Simulation Tool For Introducing Algebraic CELP (ACELP) Coding Concepts In A DSP Course,” IEEE 2002 DSP Workshop, Callaway, Georgia, October 2002

C.12 A.S. Spanias, V. Atti, Y. Ko, T. Thrasyvoulou, M.Yasin, M. Zaman, T. Duman, L. Karam, A. Papandreou, K. Tsakalis, “On-Line Laboratories For Speech And Image Processing and for Communication Systems Using J-DSP,” IEEE 2002 DSP Workshop, Callaway, Georgia, October 2002

C.13 R. Ramapriya and A. Spanias, “A Simulation Tool for introducing MPEG - Audio (MP3) concepts in a DSP course, To appear in *Proc.. IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP-2002),* Orlando, May 2002.

C.14 S. Bellofiore, J. Foutz, C. Balanis, A,S, Spanias, T, Duman, “Signal Processing and Communications Algorithms for Array Antennas, Proc.. IEEE International Symposium on Circuits and Systems (ISCAS-02), Phoenix, May 2002.

C.15 J. Foutz[+] and A. Spanias, " Adaptive Eigen-Projection Algorithms for 1-D And 2-D Antenna Arrays," *Proc.. IEEE International Symposium on Circuits and Systems (ISCAS-02),* pp. 201 –204, Phoenix, May 2002.

C.16 T. Painter and A. S. Spanias, " Sinusoidal Analysis-Synthesis of Audio using Perceptual Criteria**,”** *Proc.. IEEE International Symposium on Circuits and Systems (ISCAS-02),* Vol: 2 , pp. 177 –180, Phoenix, May 2002

C.17 Bellofiore, S.; Balanis, C.A.; Foutz, J.; Spanias, A., “Impact of smart antenna designs on network capacity “ IEEE Antennas and Propagation Society International Symposium, Volume: 3 , pp 210 –213, 2002

C.18 Salvatore Bellofiore, Jeffrey Foutz, Constantine A. Balanis and Andreas Spanias, "Smart Antennas for Wireless Communications," 2001 IEEE Antennas and Propagation International Symposium, Boston, Massachusetts, vol. 4, pp. 26-29, July 2001.

C.19 T. Painter and A. S. Spanias, "Perceptual segmentation and component selection in compact sinusoidal representations**,”** *Proc.. IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP-2001),* Salt Lake City, May 2001.

C.20 A. S. Spanias and Fikre Bizuneh, " Development of new functions and scripting capabilities in java-dsp for easy creation and seamless integration of animated dsp simulations in web courses**,”** *Proc.. IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP-2001),* Salt Lake City, May 2001.

C.21 Jeff Foutz[+] and Andreas Spanias, “Adaptive Modeling and Control of Smart Antennas,” pp. 859-862, *Proc. MIC 2001*, Innsbruck, Feb 19-22, 2001

C.22 Salvatore Bellofiore, Jeff Foutz, Israfil Bahceci, Constantine A. Balanis, Andreas Spanias, Tolga Duman and James T. Aberle, "Smart Antennas for Mobile Platforms," International Union of Radio Science, URSI 2001, Boulder, Colorado, pg. 243, Jan. 2001.

C.23 S. Ahmadi^{+} and A. Spanias, Minimum Variance Phase Prediction and Frame Interpolation Algorithms for Low Bit Rate Sinusoidal Speech Coding , pp. 345-349, *IEEE International Symposium on Circuits and Systems 2000 (ISCAS-00)*, Session P2-P1, Geneva , Switzerland, May 28-31, 2000

C.24 A. Spanias, S. Urban, A. Constantinou^{+}, M. Tampi^{+}, A. Clausen^{+}, X. Zhang^{+}, J. Foutz^{+} and G. Stylianou^{+}, "Development and Evaluation of a Web-Based Signal and Speech Processing Laboratory for Distance Learning," *Proc. IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP-2000),* Istanbul, June 2000.

C.25 Andreas Spanias, Argyris Constantinou^{+}, Jeff Foutz^{+} and Fikre Bizuneh^{+}, "An on-line signal processing laboratory," *IEEE DSP Education Workshop*, Hunt, Texas October 15-18, 2000

C.26 K. Daroudi^{+} and A. Spanias, “Frequency Selective Adaptive Modeling,” pp. 333-337, *Proc MIC 2000*, Innsbruck, Feb 14-17, 2000

C.27 Hiren Bhagatwala^{+}, Edward Painter^{+} and Andreas Spanias,"An Interactive GUI-based Tool for Signal and Speech Processing Courses," *Proceedings of ICASSP-99,* Phoenix, March 1999

C.28 Axel Clausen^{+} and Andreas Spanias,An Internet-based Computer Laboratory for DSP Courses," *Proceedings of the ASEE/IEEE Frontiers in Education Conference*, Tempe, November 1998

C.29 Xavier Anand^{+}, Andreas Spanias, and Ted Painter$^{+,}$ "An Adaptive System Identification Java Simulation for Internet based Software, *Proceedings of the ASEE/IEEE Frontiers in Education Conference*, Tempe, November 1998

C.30 A. Clausen^{+} **,**A. Spanias, A. Xavier^{+}, M. Tampi^{+}, “A Java Signal Analysis Tool for Signal Processing Experiments, *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP-98)*, DSP 16, Seattle, May 1998

C.31 A. Xavier^{+} and A.. Spanias, “An Adaptive System Identification Java Simulation for Internet based courseware, *17th International Conference on Modeling, Identification, and Control*, Grinderwald, Feb. 1998

C.32 G. Nair^{+} and A.. Spanias, “Eigenspace Projections for FIR System Identification,” *17th International Conference on Modeling, Identification, and Control*, Grinderwald, Feb. 1998

C.33 S. Ahmadi^{+}, Andreas S. Spanias, "New Algorithms for Sinusoidal Speech Coding at Low Bit Rates," *IEEE International Conference on Personal Wireless Communications*", pp. 57-61, December 1997

C.34 K. Darroudi^{+} and A. Spanias, "Robust Speech Coding based on Pole-Zero representations and Trellis Coded Quantization**,"** *The International Conference on Signal Processing Applications & Technology,* pp. 1709-1713, San Diego, September 1997

C.35 S. Ahmadi^{+} and Andreas Spanias, "A New Phase Model for Sinusoidal Transform Coding of Speec Signals**,"** *Proceedings of IEEE Mediterranean Conference on Control and Systems (CCS),* Cyprus, July 1997

C.36 T. Painter^{+}, A. Spanias, "A Review of algorithms for Perceptual Coding of Digital Audio Signals," *Proceedings of International Conference on Digital Signal Processing (DSP),* pp. 179-205, July 1997

C.37 M. Deisher^{+} and A. Spanias, "HMM - Based Speech Enhancement using Harmonic Modeling," *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP-97),* pp. II-1175, Munich, April 1997

C.38 S. Ahmadi^{+} and A. Spanias, “A New Sinusoidal Phase Modeling Algorithm," *IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP-97),* pp. III-1675, Munich, April 1997

C.39 E. Painter+ and A. Spanias, “A Matlab tool for the evaluation of Speech Coding Algorithms,” IEEE *International Conference on Acoustics Speech and Signal Processing* , Atlanta, May 1996

C.40 K. Kitsios+, A. Spanias, B. Welfert, and P. Loizou+, “An Adaptive Modified Covariance Algorithm for Spectral Analysis,” *IEEE Workshop on Statistical Signal and Array Signal Processing*, pp. 56-59, June 1996 (*)

C.41 S. Ahmadi+ and A. Spanias, “Low Bit-Rate Speech Coding Based on Harmonic Sinusoidal Models,” *International Symposium on Digital Signal Processing*, pp. 165-169, London, July 1996 (*)

C.42 P. Loizou+, A. Mekkoth+, and A.S. Spanias, “Telephone Alphabet Recognition for Name Retrieval Applications," *Proceedings of International Conf. on Signal Processing Applications and technology*, pp. 2014-2018, Boston, October 1995.

C.43 G. Tucker+, A.S. Spanias, and P. Loizou+, "An HMM-based Endpoint Detector for Computer Communication Application," *Proceedings of International Conf. on Signal Processing Applications and technology* , pp. 1969-1973, Boston, October 1995.

C.44 Min-Tau Lin+, A.S. Spanias, and F. Tiong+, "Robust Speech Recognition Based on Minimum Error Classification and Weighted Projection Measure," *Proceedings of International Conf. on Signal Processing Applications and technology (ICSPAT)*, pp. 2006-2009, Boston, October 1995.

C.45 Edward Painter+ and A.S. Spanias, "A Software Tool for Understanding and Evaluating Standardized Speech Coding Algorithms," *International Conference on Digital Signal Processing (DSP 1995)*, pp. 850-857, Limassol, June 1995.

C.46 P. Loizou+ and A.S. Spanias, "Improved Speech recognition Using the Weighted Average Divergence Measure," *International Conference on Digital Signal Processing (DSP 1995)*, pp. 90-95, Limassol, June 1995.

C.47 K. Kitsios+, A.S. Spanias, and B. Welfert, "Optimum Block Modified Covariance Algorithms for Spectral Analysis," *3rd Mediterranean Symposium on New Directions in Control and Automation*, pp. 398-405, Limassol, July 1995 (*)

C.48 G. Nair+ and A.S. Spanias, "Fast Adaptive Algorithms Using Eigenspace Projections,*" 1994 Asilomar Conference*, pp. 1520-1524, Monterey, October 1994.

C.49 M. Deisher+ and A.S. Spanias, "Speech Enhancement using a State-Based Transform Model," *1994 Asilomar Conference*, pp. 1242-1246, Monterey, October 1994.

C.50 K. Daroudi+ and A.S. Spanias, "Speech Intelligibility Improvement using the Intel Technique," *Proceedings of International Conf. on Signal Processing Applications and technology* , pp. 114-117, Dallas, October 1994.

C.51 P. Loizou+ and A.S. Spanias, "Context-dependent Modeling in Alphabet Recognition," *IEEE Proceedings of the International Symposium of Circuits and Systems (ISCAS-94)*, pp. 189-192, London, May 1994.

C.52 S. Karkada+, C. Chakrabarti and A.S. Spanias, "High Sample rate Architectures for Block Adaptive Filters,*" IEEE Proceedings International Symposium of Circuits and Systems (ISCAS-94)*, pp. 131-134, London, May 1994.

C.53 R. Fulchiero+ and A.S. Spanias, "Speech Enhancement Using the Bispectrum,*" IEEE Proc. International Conference on Acoustics Speech and Signal Processing (ICASSP-93)*, pp. 488-491, Minnesota, March 1993.

C.54 Ines Jebali+, P. Loizou+, and A.S. Spanias, "Speech Processing Using Higher Order Statistics,*" IEEE Proceedings of the International Symposium of Circuits and Systems (ISCAS-93*), pp. 160-163, Chicago, May 1993.

C.55 A. Spanias and R. Fulchiero+, "Speech Enhancement using the Least Squares Bispectrum Reconstruction Algorithm," *IEEE Proceedings of the International Conference on DSP and CAES*, pp. 434-441, Nicosia, Cyprus, 1993.(*)

C.56 K. Tsakalis, M. Deisher$^+$, A. Spanias, "System Identification Based on Bounded Error Constraints," *Proceedings of the International Conference on DSP and CAES*, pp. 75-84, Nicosia, Cyprus, 1993.

C.57 A. Elia$^+$, C. Pattichis, W. Fincham, A. Spanias, and L. Middleton, "Autoregressive Spectral Modeling of Motor Unit Action Potentials: Preliminary Findings," Proc. IEEE Intern. Conf. of Engineering in Medicine and Biology, pp. 1465-1466, Paris-France, Oct. 1992.

C.58 Q. Shen$^+$ and A.S. Spanias, "Frequency-Domain Adaptive Algorithms for Active Sound Control," *International Noise and Vibration Control Conference*, pp. 207-212, St. Petersburg, Russia, May 31-June 3, 1993.

C.59 Q. Shen$^+$ and A.S. Spanias, "An Optimal Block Adaptive Algorithm For Active Control of Sound," *Noise Control 1993 (NOISE-CON 93)*, pp. 231-236, Williamsburg, Virginia, May 2-5, 1993.

C.60 Q. Shen$^+$ and A.S. Spanias, "Frequency-Domain Adaptive Algorithms for Multi-Channel Active Sound Control," *Second Conf. on Recent Advances in Active Control of Sound and Vibration*, pp. 755-766, Blacksburg, April 28-30, 1993.

C.61 Q. Shen$^+$ and A.S. Spanias, "A Multichannel Block Adaptive Algorithm For Active Noise Control," 1992 *International Congress on Noise Control Engineering (Inter-Noise 92)*, pp. 353-356, Toronto, July 20-22, 1992.

C.62 A.S. Spanias, G. Lim$^+$, P. Loizou$^+$ and M. Deisher$^+$, "Block Modified Covariance Algorithm," *Proc. International Conference on Acoustics Speech and Signal Processing (ICASSP-92)*, Vol. 5, pp. 529-532, San Francisco, March 1992. (*)

C.63 Q. Shen$^+$ and A.S. Spanias, "Time and Frequency Domain X-Block LMS Algorithms for Single Channel Active Noise Control," *Second International Congress on Recent Developments in Air- and Structure-Borne Sound and Vibration*, Congr. Proc., pp. 353-360, Auburn, March 4-6, 1992.

C.64 A.S. Spanias, "A Hybrid Transform Method for Speech Analysis and Synthesis," *Proc. IEEE International Global Telecommunications Conference (GLOBECOM-91)*, pp. 719-724, Phoenix, Dec. 1991. (*)

C.65 M. Deisher$^+$ and A.S. Spanias, "Adaptive Noise Cancellation Using the Fast Optimal Block Algorithm (FOBA)," *Proc. IEEE International Symposium on Circuits and Systems (ISCAS-91)*, pp. 698-701, Singapore, June 1991.

C.66 P. Loizou$^+$ and A.S. Spanias, "Low rate Speech Representation by Vector Quantizing Transform Components," *Proc. IEEE International Symposium on Circuits and Systems (ISCAS-91)*, pp. 320-323, Singapore, June 1991.

C.67 P. Loizou$^+$ and A.S. Spanias, "Vector Quantization of Principal Spectral Components for Speech Coding at 1200 BPS," *IEEE Proc. International Conference on Acoustics Speech and Signal Processing (ICASSP-91)*, pp. 245-248, Toronto, May 1991.

C.68 M. Deisher$^+$ and A.S. Spanias, "Real-time implementation of a frequency domain adaptive filter on a fixed point signal processor," *IEEE Proc. International Conference on Acoustics Speech and Signal Processing (ICASSP-91)*, pp. 2013-2016, Toronto, May 1991.

C.69 P. Loizou$^+$ and A.S. Spanias, "Vector Quantization of Principal Spectral Components for Speech Coding at 4800 BPS," Presented at the *24Th. Asilomar Conference on Circuits, Systems and Computers,* Asilomar Conf. Rec., Pacific Grove, California, Nov. 1990.

C.70 A.S. Spanias, "A Hybrid Model for Speech Synthesis," *IEEE International Symposium on Circuits and Systems (ISCAS-90)*, Conf. Proc. ISCAS-90, Vol. 2, pp. 1521-1524, New Orleans, May 1990 .(*)

(*) Presentations by Andreas Spanias

C.71 A.S. Spanias, S.B. Jonsson+ and S.D. Stearns, "Transform Coding Algorithms for Seismic Data Compression," *IEEE International Symposium on Circuits and Systems (ISCAS-90)*, Conf. Proc. ISCAS-90, Vol. 2, pp. 1573-1576, New Orleans, May 1990.

C.72 W.B. Mikhael and A.S. Spanias, "A Least-Squares Pole-Zero Algorithm in the frequency and Walsh Domains with applications to speech representation," *IEEE International Symposium on Circuits and Systems (ISCAS-90)*, Conf. Proc. ISCAS-90, Vol. 2, pp. 1331-1334, New Orleans, May 1990.(*)

C.73 W.B. Mikhael and A.S. Spanias, "Direct Coding of a Class of Non-Stationary Signals Based on Mixed Transforms," *IEEE International Symposium on Circuits and Systems (ISCAS-89)*, Conf. Proc. ISCAS-89, Vol. 1, pp. 280-283, Portland, May 1989.(*)

C.74 W.B. Mikhael and A.S. Spanias, "Fourier-Walsh Representation of a Class of Non-Stationary Signals," *IEEE International Symposium on Circuits and Systems (ISCAS-89)*, Conf. Proc. ISCAS-89, Vol. 3, pp. 1768-1771, Portland, May 1989.(*)

C.75 W.B. Mikhael and A.S. Spanias, "Reduced Bit-rate Representation of Speech Using Mixed Fourier-Walsh Transforms," *22nd. Asilomar Conference on Circuits, Systems and Computers*, Pacific Grove, California, pp. 366-370, Nov. 1988. (*)

C.76 W.B. Mikhael, A.S. Spanias, F.H. Wu, "Fast Frequency-Domain Implementation of a Block IIR Filter with Applications," *IEEE International Symposium on Circuits and Systems (ISCAS-88)*, Conf. Proc. ISCAS-88, pp. 285-288, Espoo, Finland, June 1988.(*)

C.77 A.S. Spanias and W.B. Mikhael, "Implementation of the Optimum Block Adaptive Algorithm in the Frequency-Domain," *IEEE International Symposium on Circuits and Systems (ISCAS-87)*, Conf. Proc. ISCAS-87, pp. 426-429, Philadelphia, May 1987. (*)

C.78 W.B. Mikhael and A.S. Spanias, "Performance Enhancement of the Frequency-Domain LMS Adaptive Algorithm," *IEEE International Symposium on Circuits and Systems (ISCAS-86)*, Conf. Proc. ISCAS-86, Vol. 1, pp. 349-352, San Jose, California, May 1986. (*)

C.79 W.B. Mikhael, A.S. Spanias, F.H. Wu, "An Adaptive Pole-Zero Predictor," *IEEE International Symposium on Circuits and Systems (ISCAS-85)*, Conf. Proc. ISCAS-85, Vol. 3, pp. 1107-1110, Kyoto, Japan, June 1985.

C.80 W.B. Mikhael, A.S. Spanias, and F.H. Wu, "ARMA Modeling Using the Linear Predictor," *International Conference on Acoustics Speech and Signal Processing (ICASSP-85)*, Conf. Proc. ICASSP-85, Vol. 4, pp. 1505-1508, Tampa Bay, Florida, April 1985. (*)

C.81 K. Tsakalis, M. Deisher, A. Spanias+, "FIR Adaptive Filtering Based on a Bounded Error Criterion," *Proc. Asilomar Conference on Circuits, Systems and Computers*, **Invited**, pp. 15-18, Monterey, Nov. 1992.

C.82 A.S. Spanias and F.H. Wu, "Speech Coding and Speech Recognition Technologies: A Review," *Proc. IEEE International Symposium on Circuits and Systems (ISCAS-91)*, **Invited,** pp. 572-577, Singapore, June 1991. (*)

C.83 A.S. Spanias and F.H. Wu, "Speech Coding and Recognition: A Review," *Proc. of the First Cyprus International Conference on Computer Applications to Engineering Systems*, **Invited,** pp. 46-71, July 1991.(*)

C.84 S.B. Jonsson+ and A.S. Spanias, "Seismic Data Compression," *IEEE International Phoenix Conference on Computers and Communications (IPCCC-90)*, Conf. Proc., **Invited,** pp. 276-279, Phoenix, March 1990.(*)

C.85 A.S. Spanias and W.B. Mikhael, "An Adaptive Bit Allocation Scheme for Coding of Speech Signals Using Partial Sets of Orthogonal Functions," *32nd Midwest Symposium on Circuits and Systems (MWCAS-89)*, Conf. Proc. MWCAS-89, **Invited,** Session TAM4-S, Champaign, Illinois, August 1989. (*)

C.86 W.B. Mikhael and A.S. Spanias, "Representation of Speech Signals using Mixed Incomplete Sets of Basis Functions," *21st. Asilomar Conference on Circuits, Systems and Computers*, Conf. Rec., Vol. 2, **Invited** , pp. 905-908, Pacific Grove, California Nov. 1987. (*)

C.87 W.B. Mikhael, A.S. Spanias and F.H. Wu, "ARMA modeling by cascading a Linear Predictor and a Pole-Zero Structure," *18Th. Asilomar Conference on Circuits, Systems and Computers*, Conf. Rec., **Invited,** pp. 63-70, Pacific Grove, California, Nov. 1984.

C.88 A. Clausen^{+}, T. Painter^{+}, A. Xavier^{+}, M. Tampi^{+}, T. Lam^{+}, A. Constantinou^{+}, and A. Spanias, “J-DSP: An Internet-based Educational Tool for Digital Filter Experiments, *1998 IEEE Symposium on Advances in Digital Filtering and Signal Processing,* **Invited**, Victoria, BC, pp. 57-61, June 5-6, 1998

C.89 A. Spanias and T. Painter^{+}, “Network Applications of Speech and Audio Coding Algorithms,” Article in "Multimedia Over the Broadband Network: Business Opportunities and Technologies," **Invited**, International Engineering Consortium, June 1996.

C.90 A.S. Spanias, "A Pole-Zero Adaptive Algorithm for Speech Processing," *IEEE International Phoenix Conference on Computers and Communications (IPCCC-90),* **Invited,** Conf. Proc., Page 894, Phoenix, March 1990 (*).

C.91 A.S. Spanias, M. Pattichis, M. Souropetsis, D. Petrondas, A. Schizas, L. Middleton, "Linear Prediction Analysis Applied in EMG," C. Pattichis, *IIId International Conference on Quantitative EMG*, **Invited,** page 35, Larnaca, Cyprus, June 1988.

Non-Refereed Editorials and Abstracts

"A Block Modified Covariance Algorithm for Spectral Analysis," A.S. Spanias, IEEE Transactions on Signal Processing, Vol. 48(2), p. 2123, Aug. 1992

A. Spanias, “Signal Processing Society Conferences; Vice President Column,” IEEE *Signal Processing Magazine*, editorial, Sept. 2001.

Book / Software Publishing (submitted and under review)

- Perceptual Audio Coding, Ted Painter and Andreas Spanias, Prentice Hall, To be published, Dec 2003
- Speech Coding for Mobile and Multimedia Applications, Andreas Spanias, In final Preparation
- Java DSP Software Concept, ~33,000 lines, ISBN 0-9724984-0-0, Copyright © A. Spanias, 2002.

Invited Contributions in Books - Book Chapters

- Andreas Spanias, Chapter 3: Speech Coding Standards, pp. 25-44, **Invited.** Academic Press, Ed: G. Gibson, ISBN 2000 0-12- 282160-2

- A. Spanias, Speech Coding for Mobile and Multimedia Applications, **Invited** Chapter in "Digital Signal Processing Technologies-Critical Technology Reviews (CR57)," Eds. P. Papamichalis and R. Kerwin, pp. 115-144, SPIE Press, Washington, 1995

- A. Spanias, Speech Coding for Wireless Applications, **Invited** Chapter in "Microsystems Technology for Multimedia Applications," Eds. Sheu et al, Chapter 4, pp. 257-279, IEEE Press, New York 1995

Other Publications

Scientific Reports - Technical Reports for Funded Research Projects

R.1. "Smart Antennas for Mobile Networks," Progress Report Submitted to the National Science Foundation, C. Balanis, A. Spanias, et al, May 2000

R.2. "Universal Speech and Audio Coding Using a Sinusoidal Signal Model," A. Spanias, S. Ahmadi, and T. Painter, To Intel MRC, ASU-TRC Technical Report TRC-SP-ASP-9603, September 1996

R.3. "Design, Analysis, and Implementation of the GSM Channel Codec and Modem," A. Spanias, J. Sadowsky, K. Daroudi, A. Mekkoth, S. Azizi, To Intel MRC, ASU-TRC Technical Report TRC-SP-ASP-9602, August 1996

R.4. "A Software Tool for the Evaluation of Speech Enhancement Algorithms on the Intel Pentium Processor," A. Spanias and M. Deisher, To Intel MRC, ASU-TRC Technical Report TRC-SP-ASP-9501, May 1995

R.5. "Preprocessing Algorithms for Speech Recognition," A. Spanias, P.Loizou, and G. Tucker, ASU-TRC Technical Report TRC-SP-ASP-9403, May 1994.

R.6. "Speech Coding Algorithms for Mobile Communications: A Review," A. Spanias, ASU-TRC Technical Report TRC-SP-ASP-9402, April 1994.

R.7. "Speech Enhancement for Mobile Communications," A. Spanias and M. Deisher, ASU-TRC Technical Report TRC-SP-ASP-9401, Reported to Intel Corporation, April 1994.

R.8. "Implementation of the VSELP Algorithm on the EP+: Addendum to Fixed-point Implementation of the VSELP Algorithm," A. Spanias and M. Deisher, ASU-TRC Technical Report TRC-SP-ASP-9305, Reported to Intel Corporation, October 1993.

R.9. "Speech Enhancement Using the Pseudocepstrum: Final Report," A. Spanias and K. Daroudi, ASU-TRC Technical Report TRC-SP-ASP-9304, Reported to Motorola Inc., June 1993.

R.10. "Design, Analysis, and Implementation of the GSM Modem," A. Spanias, Y. Zhang, and F. Tiong, ASU-TRC Technical Report TRC-SP-ASP-9303, Reported to Intel Corp., March 1993.

R.11. "Active Noise Cancellation in Ducts - Final Report," A. Spanias and J. Liu, ASU-TRC Technical Report TRC-SP-ASP-9302, Reported to ANVT, March 1993.

R.12. "Speech Enhancement Using the Pseudocepstrum: Task 3," A. Spanias and K. Daroudi, ASU-TRC Technical Report TRC-SP-ASP-9301, Reported to Motorola Inc., March 1993.

R.13. "Speech Enhancement Using the Pseudocepstrum: Task 2," A. Spanias and K. Daroudi, 50 pages, ASU-TRC Technical Report TRC-SP-ASP-9206, Reported to Motorola Inc., November 1992.

R.14. "Analysis and Implementation of the GSM RPE-LTP Algorithm," A. Spanias, P. Loizou, and G. Lim, 48 pages, ASU-TRC Technical Report TRC-SP-ASP-9205, Reported to Intel Corp., October 1992.

R.15. "Active Noise Cancellation in Ducts - 3rd Quarter," A. Spanias and J. Liu, 12 pages, ASU-TRC Technical Report TRC-SP-ASP-9204, Reported to ANVT, September 1992.

R.16."Speech Enhancement Using the Pseudocepstrum: Task 1," A. Spanias and K. Daroudi, 13 pages, ASU-TRC Technical Report TRC-SP-ASP-9203, Reported to Motorola Inc., August 1992.

R.17."Simulation Models for the GSM Channel Coding and Modem Functions," A. Spanias, W. Ma, and F. Tiong, 25 pages, ASU-TRC Technical Report TRC-SP-ASP-9202, Reported to Intel Corp., August 1992.

R.18."Fixed Point Implementation of the VSELP algorithm: Final Report," A. Spanias, M. Deisher, P. Loizou, and G. Lim, 286 pages, ASU-TRC Technical Report TRC-SP-ASP-9201, Reported to Intel Corp., July 1992.

R.19."Development and Evaluation of Fixed-Point Full- and Half-Rate GSM Coders: Progress Report on the Full Rate GSM Speech Codec: 3rd Quarter" A. Spanias, P. Loizou, G. Lim, and M. Deisher, 14 pages, ASU Report CRR 92070, Reported to Intel Corp., June 1992.

R.20."Fixed Point Implementation of the VSELP algorithm: Task 3," A. Spanias, M. Deisher, P. Loizou, and G. Lim, 30 pages, ASU Report CRR 92069, Reported to Intel Corp., June 1992.

R.21."Active Noise Cancellation in Ducts - 2nd Quarter," A. Spanias and **J. Liu**, 20 pages, ASU Report CRR 92066, Reported to ANVT, June 1992.

R.22."Development and Evaluation of Fixed-Point Full- and Half-Rate GSM Coders: Progress Report on the Full Rate GSM Speech Codec: 1st & 2nd Quarter" A. Spanias, P. Loizou, and G. Lim, 19 pages, ASU Report CRR 92057, Reported to Intel Corp., March 1992.

R.23."Fixed Point Implementation of the VSELP algorithm: Phase 2, Tasks 1 and 2," A. Spanias, M. Deisher, P. Loizou, and G. Lim, 18 pages, ASU Report CRR 92045, Reported to Intel Corp., March 1992.

R.24."Active Noise Cancellation in Ducts," A. Spanias and J. Liu, 21 pages, ASU Report CRR 92044, Reported to ANVT, February 1992.

R.25."Fixed Point Implementation of the VSELP algorithm: Phase 1," A. Spanias, M. Deisher and P. Loizou, 14 pages, ASU Report CRR 92001, Reported to Intel Corp., June 1991.

R.26.Transform Coding for Seismic Data Compression," A.S. Spanias and S.B. Jonsson, 104 pages, ASU Report CRR-91001, Reported to Sandia National Labs for contract No. 54-0899, July 1990.

Patents

“Phase Compensation for Sinusoidal Transform Coding of Audio Signals,” S. Ahmadi and A. Spanias, Arizona State University, Submitted

Invited Presentations

- University of Cyprus, Speech and Audio Coding Technologies, Combined IEEE and Department of Computer Science, Nicosia, Cyprus, March 12, 2002
- General Dynamics, Vocoder Technologies for Secure Communications, Signal Processing Excellence Series, Scottsdale AZ, April 19, 2002
- "Smart Antennas," TRC Industry Advisory Board, Feb. 2001
- "An on-line signal processing laboratory," IEEE DSP Education Workshop, Hunt, Texas, October 15-18, 2000

- “Speech Coding,” University of Manchester, April 1999
- “Speech Processing,” ASU Systems Science/TRC Seminar, Tempe, February 1996.
- “Sinusoidal Models for Speech Coding,” Intel Corporate Research Council, Portland, February 1996.
- “Speech Coding,” Imperial College, London, June 1995
- Speech Coding Algorithms, Texas Instruments Corporate Communications Group, Dallas, September 1995
- "On the Utility of the FFT in electromyography," ASU ESPE Department - - Biomechanics Lab, March 28, 1995
- "Speech Coding Technologies," IEEE Communications and Signal Processing Phoenix Chapter, December 6, 1994,
- Several invited conference paper presentations by A.S. Spanias marked with (*) in the list of conference publications.
- "System Identification Algorithms," Presentation to the Du Pont Process Control Technologies Panel, September 16, 1992
- "Speech Compression Algorithms," Presentation to the Connectivity Group at Intel Corporation, August 14, 1992
- "Speech Coding," Seminar given on February 13, 1990, Systems Science and Engineering, ASU
- "Speech Coding for PCN," Lecture given on March 10, 92, Telecommunications Research Center, ASU
- "Speech Processing Algorithms: Development, Analysis, and Evaluation," April 21, 1992, Speaker at the Motorola Group (SABA Meeting) at the Center for Professional Development, ASU
- "Spectral Analysis of EMG Signals," July 1, 1992, Seminar given at the Cyprus Institute of Neurology and Genetics (CING).

RESEARCH GRANTS AND CONTRACTS

External Research Grants and Contracts

1. PI: Andreas Spanias, CO-PIs T. Duman, A. Papandreou, K. Tsakalis, L. Karam, "Java DSP - Extensions to Communications Advanced DSP, Controls, Image, **NSF**, JRA-0001, $ 424,770, Jan 2001-Jan 2004

2. CO-PI; A.S. Spanias, PI: C. Balanis and 4 other CO-PIs, **NSF**, "Smart Antennas," $458,100, Sept. 2000 - Aug. 2002.

3. PI; A.S. Spanias, **Intel Corp.**, "Distributed Voice Recognition System for the PC," $58,100, DWT0018, Sept. 1996 - Jan. 1998.

4. PI: A.S. Spanias and CO-PI: J. Sadowsky, Analysis and Implementation of CDMA Mobile Communications, Amount: $241,457.00, **Intel Corp.,** DWT 0011, Aug. 1996-Aug. 1997.

5. PI: A.S. Spanias, Development of Universal and Interoperable Speech and Audio Compression Algorithms for Multimedia and Teleconferencing Applications, Sponsor**: Intel Corp**., Amount: $177,354, DWT 4598, Feb. 1995-Jan. 1998.

6. PI: A.S. Spanias and CO-PI: J. Sadowsky, Implementation and Integration of the Speech Codec, Channel Coder/Decoder, and Signaling Protocol on Prototype DSP Chips**, Intel Corp.**, Amount: $243,500.00, DWT 4630, May. 1995-Aug. 1996.

7. PI: Chaitali Chakrabarti, CO-PI: A.S. Spanias, "Special Purpose Architectures for Speech Coding Algorithms-Phase 2," Sponsor: Motorola Inc, $15,000, Aug 16, 1995- Aug 14, 1996.

8. PI: A.S. Spanias, Analysis and Implementation of Modem Algorithms on Intel DSP Architectures, Sponsor: **Intel Corp.**, Amount: $56,939.00, Aug. 1994-Feb. 1995.

9. PI: A.S. Spanias, Speech Enhancement Algorithms for Mobile Communications, Sponsor: **Intel Corp**., Amount: $37,940.00, DWT 4460, Aug. 1994-Aug. 1995.

10. PI: A.S. Spanias, CO-PI: C. Chakrabarti, Speech Coding Algorithms for Multimedia Applications, Sponsor: **Intel Corporation**, Amount: $54,728, Sept. 1993-Aug. 1994.

11. PI: Chaitali Chakrabarti, CO-PI: A.S. Spanias, "Special Purpose Architectures for Speech Coding Algorithms," Sponsor: Motorola Inc, $15,745.00., May 16, 1994- May 14, 1995.

12. PI: A.S. Spanias, Image Processing Algorithms for Teleconferencing and Multimedia Applications, Sponsor: **Motorola Inc**., Amount: $45,000, Feb. 1 1994-Jan. 31 1995.

13. PI: A.S. Spanias, Development of Speech Encoding and Recognition Algorithms for the Phoenix Architecture: Phase 2, Sponsor: **Intel Corp**., Amount: $200,229.00, Aug. 1993-Aug. 1994.

14. PI: A.S. Spanias, Speech Enhancement Algorithms for Mobile Communications, Sponsor: **Intel Corp**., Amount: $36,130.00, Aug. 1993-Aug. 1994.

15. PI: A.S. Spanias, Development of Speech Encoding, Recognition, and Data Encryption Algorithms for the Phoenix Architecture, Sponsor: **Intel Corp**., Amount: $192,781.00, CRP 92373, DWT 4473, Aug. 1992-Dec. 1993.

16. PI: A.S. Spanias, CO-PI: Jennie Si, Performance Evaluation of Voice Recognition Algorithms, Sponsor: **Motorola Inc**, $19,845.00., February 1993-July 1993.

17. PI: A.S. Spanias, Enhancement of Speech Using the Pseudocepstrum, Sponsor: **Motorola GEG,** $39,955.00., CRP 92265, DWT 4460, February 1992-February 1993.

18. PI: A.S. Spanias, Development and Evaluation of Fixed-Point Full and Half-Rate GSM Coders, Sponsor**: Intel Corp**., Amount: $233,463.00, CRP 92079, DWT 4432, Date: September 1991-December 1992

19. PI: A.S. Spanias, "Active Noise Cancellation in Ducts, "Sponsor: Active Noise and Vibration Technologies, Amount: $27,682.00, CRP 92039, DWT 8504, Date: August 1991-December 1992

20. PI: A.S. Spanias, Fixed Point Implementation of the VSELP algorithm, Sponsor: Intel Corp., Amount: $55,984.00, CRP 91289, DWT 4423, Date: May 1991-June 1992

21. PI: A.S. Spanias, "Transform Coding for Seismic Data Compression," Sponsor: Sandia National Laboratories (SNL), CRP 90009, $19,982.00, DWJ 6150, November 1989-October 1990.

22. PI: A.S. Spanias (and overall project director) and 13 other CO-PIs from four different colleges (CEAS, CLAS, COE, and CEE), "Multidisciplinary Research on Multimedia Technologies for Distributed Learning Using the Intel PC and the Internet, $67,000, **Intel Corporation**.

23. PI: D. Evans, CO-PI: A.S. Spanias and 9 other CO-PIs (A. Spanias is one of three directors on this project), Planning Grant for a Center for Collaborative Research in Learning Technologies, **NSF**, $ 50,000.

24. PI: A. Spanias and 15 other CO-PIs, "Multidisciplinary Research on the Next Generation Multimedia Technologies for Interactive Distributed Learning," **State of Arizona, ASU VPR Multidisciplinary Initiative Committee**, Pre-proposal already approved, $ 150,000 for three years.

External Funded Equipment Proposals

- PI: A.S. Spanias, For the Development of Speech Coding Algorithms for Teleconferencing Applications, Two Intel Pentium Multimedia PCs, Amount: $12,500.00, Date: Feb. 1995.

- PI: A.S. Spanias, Various DSP boards donated to A. Spanias after proposal by Motorola, AT&T, Intel , DSP Group boards, Totaling Approximately : $34,254.00, Dates: 1992-1994.

Internal Funded Research Projects

- PI: A.S. Spanias, “Multimedia Education over the Internet,” CIEE FGIA, May 1996- May 1997, $6,000.

- PI: A.S. Spanias, "Speech Processing Based on Higher Order Statistics (HOS)," DWR B708, Agent: Faculty Grant-In-Aid Program; ASU, January 1991-December 1991, $5,000.00.

- PI: A.S. Spanias, "Low Bit-Rate Speech Coding," DWR B575, Agent: Faculty Grant-In-Aid Program; ASU, January 1989-December 1989, $3,000.00.

Internal Awards and Equipment Support Funds

- Funds for Equipment Upgrades and Acquisitions in the Speech Processing Lab, Amount $32,000 (Engineering College, EE, TRC), 1995-96
- Funds for Equipment on the Project: Development and Evaluation of Fixed-Point Full and Half-Rate GSM Coders, PI: A.S. Spanias, Amount: $36,000.00 (Funds from ASU VPR, EE, TRC), Date: November 1991
- Research Incentive Award on the Project: A Hybrid Model for Speech Coding, PI: A.S. Spanias, Amount: $16,000.00 (Funds from VPR, EE, TRC), Date: August 1989.

Labs and Facilities Developed

- DSP Lab - From Funds by Donations and Equipment Grants $200k, 1989-2001
- Speech Systems Lab, Anechoic Chamber, From Funds by Donations and Equipment Grants $150k, 1989-2001
- Distance Learning Lab, From Funds by Donations and Equipment Grants $200k, 1995-2001

Other Research Proposals

- PI: A.S. Spanias, Speech Coding Libraries for Microcontrollers, Amount: $52,500.00, May. 1995-May. 1996.
- PI: A.S. Spanias, with 5 CO-PIs, Development of Signal Processing and Communications Algorithms and Design of Small Antennas for a Campus Security System, Sponsor: Motorola GSTG (Prime: ARPA), Amount: $589,806.00, Jan. 1994-Dec. 1995.
- CO-PI: A.S. Spanias with 20 others, Intelligent Vehicle / Highway Systems Research Centers of Excellence Program, Sponsor: Federal Highway Administration, Amount: $1,398,123, Aug. 1993-Aug. 994.
- CO-PI: A.S. Spanias with 24 others, "Foundations of Intelligent Systems: MOVE," Army Research Office, URI, CRP: 92013, Jan. 1992-Dec. 1995, $1,662,143.00.
- PI: A.S. Spanias, "Speech Processing Based On Higher Order Statistics," RIA-National Science Foundation, CRP: 91254, Jun. 1991-May 1993, $60,000.00.

- CO-PI: A.S. Spanias with D. Morrell, "Voice and Data Processing Algorithms for the NLOS Communications," McDonnell Douglas Helicopter, CRP: 91203, Mar. 1991- Mar. 1992, $20,332.00.

- CO-PI: A. Spanias with D. Morrell and D. Cochran, "Radar/IR Image Object Classification," Motorola GEG, CRP: 90255, Jan. 1990-Jan. 1991, $49,916.00.

- PI: A.S. Spanias, "Spectral Analysis of EMG," Agent: Institutional Biomedical Research Support Program; ASU, CRP: 91081, Jan. 1991-Dec. 1991, $8,500.00.

- PI: A.S. Spanias, "A Hybrid Model for Speech Coding," Department of Defense (DOD), CRP: 90036, Feb. 1990-Jan. 1993, $225,290.00.

- PI: A.S. Spanias, "Low-Rate Speech Coding Using Complex Spectral Functions," US West Advanced Technologies, CRP: 89164, June 1989-May 1990, $40,395.00.

Consulting

- Intel Corporation, Architecture Design, Speech Coding Algorithms
- Inter-Tel Communications: Worked on Adaptive Echo Cancellers for the Athena Project
- Motorola Inc: Worked on Speech Recognition Algorithms, 1992
- Texas Instruments, DSP Training, Sept. 1995
- The Cyprus Institute of Neurology and Genetics: Participated in Spectral Analysis of Electromyographic Signals for Automatic Diagnosis of Neuromuscular diseases.

Student Theses and Dissertations Supervised

Ph.D. Dissertation Supervision (Completed)

1. "Adaptive Algorithms for GPS systems," B. Badke, Dept. Electr. Eng., ASU, Dec. 2002

2. “Perceptual Coding of Digital Audio,” Ted Painter, Dept. Electr. Eng., ASU, August 2000. (T. Painter is with Intel)

3. “Sinusoidal Modeling of Wideband Signals,” Khosro Daroudi, Dept. Electr. Eng., ASU, December 1999. (K. Daroudi is with Intel)

4. “Adaptive Filters Based on Eigenspace Projections,” Gopal Nair, Dept. Electr. Eng., ASU, May 1998. (G. Nair is with Intel)

5. “An Improved Approach to Robust Speech Recognition Using Minimum Error Classification,” Min-Tau Lin, Dept. Electr. Eng., ASU, December 1997. (M. Lin is now with Solectron in San Jose)

6. "Low Bit Rate Coding based on the Sinusoidal Model," Sassan Ahmadi, Dept. Electr. Eng., ASU, August 1997. (S. Ahmadi is with Nokia at San Diego)

7. “State-Based Noise Reduction Using the Sinusoidal Speech Model,” Mike Deisher, Dept. Electr. Eng., ASU, May 1996 (Mike Deisher is currently with Intel Corporate Research in Portland)

8. "Robust Speaker Independent Recognition of Alphabet Symbols," Philipos Loizou, Dept. Electr. Eng., ASU, May. 1995. (Philipos Loizou is now Associate Professor at the University of Texas Dallas)

9. "Single and Multiple Channel Block Adaptive Filters for Active Noise Cancellation," Qun Shen, Dept. Electr. Eng., ASU, Dec. 1992. (Qun Shen is Ericsson at the Research Triangle Park)

M.S. Theses Supervision (Completed)

1. " MATLAB Implementation of the G. 729," V. Atti, Dept. Electr. Eng., ASU, Dec. 2002

2. "Algorithms for Beamforming," "S. Miller, Dept. Electr. Eng., ASU, Dec. 2002

3. "Analysis and Implementation of the MP-3 standard, Rama Ramapryia, Dept. Electr. Eng., ASU, Dec. 2001

4. ‘Analysis and Evaluation of G.723.1,” Mugundan Narayanan, Dept. Electr. Eng., ASU, Aug 2000 (Mugundan is now with Intel)
5. “Development of Java DSP,” Argyris Constantinou, Dept. Electr. Eng., ASU, Dec. 1999 (Argyris is now with Globalsoft)
6. “Development of Speech Processing Functions for JDSP,” Maya Tampi, Dept. Electr. Eng., ASU, Dec. 1999 (Maya is now with Motorola)
7. "Multichannel Noise Cancellation," Anand Xavier+, Dept. Electr. Eng., ASU, Aug 1998 (Anand is in Signalogic).
8. "A Comparison of Vocoders for Cellular Systems with Emphasis on the Enhanced Variable Rate Codec (EVRC)," Hiren Bhagatwala+, Dept. Electr. Eng., ASU, May 1998 (Hiren is now with Qualcom).
9. "Adaptive Modified Covariance Algorithms with Time-Varying Gains," Kyriacos Kitsios+, Dept. Electr. Eng., ASU, May 1995. (Kiriakos is now with J&P)
10. "Analysis and Implementation of Speech Coding Algorithms," Ted Painter+, Dept. Electr. Eng., ASU, May 1995. (Ted is with Intel)
11. “Endpoint Detection for Isolated Word Recognition Using Hidden Markov Models,” Greg Tucker+, Dept. Electr. Eng., ASU, May 1995. (Greg is currently with Intel)
12. "Normalized Frequency-Domain Modified Covariance Algorithms," Francis Tiong+, Dept. Electr. Eng., ASU, Dec. 1994. (Francis is currently with Phylon in San Jose, CA)
13. "High Sample Rate Architectures for the BLMS Algorithm," Karkada Srikanth, Dept. Electr. Eng., ASU, Dec. 1993. (Karkada is currently with Phillips in San Jose, CA)
14. "Speech Processing Using the Bispectrum," Ralph Fulchiero, Dept. Electr. Eng., ASU, Aug. 1993. (Ralph is currently with Motorola GSTG)
15. "Frequency-Domain All-Pole Spectral Matching With Applications to Speech," Gim Lim+, Dept. Electr. Eng., ASU, Dec. 1992. (Gim is currently with Intel)
16. "Speech Analysis and Enhancement Using Higher Order Statistics," Ines Jebali Gdoura, Dept. Electr. Eng., ASU, May. 1992. (Ines is with the Telecom company in Tunisia)
17. "Digital Image Restoration using the Pseudo-Cepstrum," Ye-Quang Chen, Dept Electr. Eng., ASU, Dec. 1991. (Ye-Quang is with the Department of Defense in Taiwan)
18. "The Discrete Wavelet Transform and its Application To Signal Reconstruction," Gen-Fuh Hu, Dept. Electr. Eng., ASU, Dec. 1991. (Gen-Fuh is in Taiwan)
19. "Experimental Analysis of Frequency-domain Adaptive Noise Cancellers," M.E. Deisher, Dept. Electr. Eng., ASU, May 1991. (Mike completed a Ph.D. at ASU is now with the Corporate Research labs at Intel)
20. "Low-rate Speech Representations by Vector Quantizing Transform Components," P. Loizou+, Dept. Electr. Eng., ASU, May 1991.
21. "Design Considerations for a 94 GHz Pulsed Doppler Radar System with Interactive Computer Processing," J.W. Nehrbass, Dept. Electr. Eng., ASU, May 1991.
22. "Transform Coding of Seismic Data," S.B. Jonsson +, Dept. Electr. Eng., ASU, May 1990. (Stefan is with Honeywell in Phoenix)
23. "Block Adaptive Prediction Algorithms," G.A. Sarrouh, Dept. Electr. Eng., ASU, May 1990. (Until 1993 George was employed in Tempe)
24. "Time and Frequency domain Adaptive Noise Cancellers," V. Rhodes, Dept. Electr. Eng., ASU, May 1990. (Val is with EF Data in Tempe)

Student Dissertations/Theses Supervision (In Progress)

- “Language Modeling for Voice Recognition,” Ajith Mekkoth, Ph.D.
- “DSP Algorithms for Smart Antennas,” Jeff Foutz, Ph.D.
- “Array Antennas for CDMA systems,” Steve Miller, Ph.D
- "Speech Modeling using microradars ", Atti Venkataraman, Ph.D+
- "Beamforming Algorithms ", Ashwin, Ph.D
- "Digital Communications", Ghassan Malouli,, Ph.D
- “Psychoacoustic models in speech coding,” Yu Song+
- “Adaptive Beamforming Algorithms ", Thrassos Thrassyvoulou, M.S+.
- “Adaptive Equalization Algorithms ", Costas Constantinou, M.S.+
- “ Communications Algorithms for Java DSP,” Fikre Bizuneh, M.S.
- “ Analysis of MP3 and its use in Noise Reduction,” Ryan Pintoi M.S.
- “ Analysis of DNA Data using Linear Prediction,” Niranjan Chakravarti, M.S. +
- "Extensions on Java DSP in Communications", Atti Venkataraman, M.S.+
- “ Microphone Beamforming,” Seth Benton, M.S. +

- + Students that have been or are being funded from sponsored research projects

Professional and Scientific Service

Local IEEE Activities

- IEEE Communications and Signal Processing (COMSOC/SP), Phoenix Chapter, **Chair**, 1993-97. (Coordinate local IEEE meetings)
- IEEE Communications and ASSP Society (COMSOC/SP), Phoenix Chapter, Vice Chair, 1990-93.

Membership in National and International Committees

- Elected Member of the IEEE Signal Processing Society Technical Committee on Statistical Signal and Array Processing (formerly Spectrum Estimation and Modeling), 1991-1997.
- Elected Member of the IEEE Signal Processing Conference Board, 1993-1999.
- Elected Member of the IEEE Circuits and Systems Society (CAS) Technical Committee on Digital Signal Processing, 1992-1997

Major Scientific Service in IEEE Signal Processing Society

- Vice-President Conferences, IEEE Signal Processing Society, 2000-2002
- Member Board of Governors, IEEE Signal Processing Society, 2000-2002
- Member Executive Committee**,** IEEE Signal Processing Society, 2000-2002
- Associate Editor**,** IEEE Signal Processing Letters, 2000-2002.
- Associate Editor**,** IEEE Transactions on Signal Processing, 1994-1997.
- **Genera**l Conference Co-Chair, (along with D. Cochran) 1999 IEEE International Conference on Acoustics Speech and Signal Processing (ICASSP-99), Phoenix, March 1999.
- Guest Co-Editor**,** IEEE_Signal Processing Magazine, Jan 2000 Special Issue on Industry Applications
- Guest Editor**,** IEEE_Signal Processing Magazine, March 2000 Special Issue on Industry DSP Technology
- Founder and Chair Industry DSP Committee**,** IEEE Signal Processing Society

Other Conference Activities

- Organizaer and Chair Mini-workshop on Signal Processing for Communications and Multimedia, Tempe, Feb. 2002
- Program Committee Melecon 2000, 10th Mediterranean Electrotechnical Conference, May 29-31, 2000, CYPRUS
- Organizer and Chair, Special Session on "Speech Coding," 1995 International Conference on DSP, Limassol, 1995.
- Program Committee and Session Chair, 1995 IEEE Midwest International Symposium on Circuits and Systems (MWCAS-95), Brazil, August 1995.
- Co-Organizer and Chair, Special Session on "Recent Advances in Speech Processing," 1991 IEEE International Symposium on Circuits and Systems (ISCAS-91), Singapore, June 1991.
- Organizer and Chair, Special Session on "Low-rate signal coding," 1990 IEEE International Phoenix Conference on Computers and Communications (IEEE IPCCC-90), Scottsdale, March 1990.
- Track Chair for the Communications Technology Sessions (6 sessions) of IEEE IPCCC-91
- Session Chair in the IEEE IPCCC-92 and IPCCC-93
- Session Chair in the 1993 International Conference on DSP and CAES, Nicosia, 1993.
- University Liaison and Program Committee member in IEEE IPCCC-92, IPCCC-93, and IPCCC-94
- Co-organizer, Special Session on "Representation of Time-Varying Signals," 21st Annual Asilomar Conference, Asilomar Conference Grounds, Pacific Grove, Nov. 1987.

Professional and Scientific Service (cont.)

Paper and Book Reviews

- Reviewed papers for several IEEE Transactions, i.e., Circuits and Systems, Signal Processing, Communications, Selected Areas in Communications, and Geoscience and Remote Sensing. Also reviewed papers for IEE Proceedings - Part I, ACM, the Journal of Adaptive Signal Processing and Control, the 1991,1992, and 1993 International Phoenix Conference on Computers and Communications (IPCCC-91, IPCCC-92, IPCCC-93), the IEEE Workshop on Statistical Signal and Array Processing, the 1993, 1994, 1995, 1996, 1999, 2000, 2001 IEEE International Conference on Acoustics, Speech, and Signal Processing, the 1996 IEEE International Symposium on Circuits and Systems.
- Reviewed Books : "Digital Signal Processing" by Mitra (1996), Papoulis' 2nd Edition Signals and Linear Systems book for McGraw Hill (1995), Higher Order Statistics for Prentice Hall, in 1993.
- Reviewed NSF Proposals

Short Courses and Tutorials

Developed and taught short courses devoted to continuing engineering education in the areas of Digital and Adaptive Signal Processing and Speech Coding. These are:

- Fundamentals of Digital Signal Analysis. This three-day (lecture and computer laboratory) course was given on July 31st, 1990 (at Makarios Hospital, Nicosia-Cyprus), January 7th, 1991 (at ASU), September 16, 1991 (at ASU), November 4, 1992 (at Motorola GEG), April 23, 1993 (at ASU), March 14-16, 1994 (at ASU), September 1995 (Texas Instruments). June 1996, 1997, 1998, 1999, 2000 (Phoenix). Support material: 400 pages.

- Fundamentals of Adaptive Signal Processing. This course (8 hours) was sponsored by the Industrial Training Authority of Cyprus in association with the Cyprus Institute of Neurology and Genetics. This one-day course was given twice at the Makarios Hospital, Nicosia-Cyprus on August 5, 1991 and June 17, 1992. Support material: 200 pages.

- Speech Coding for Mobile and Multimedia Applications. This course (16 hours) was sponsored by the Professional Development Center of Arizona State University (ASU) and was held March 17-18, 1994. An expanded version (24 hours) of this course was also given May 18-20, 1994 to Intel engineers in Chandler. Also held June 1995 and June 1996, 1997, 1998, 1999, 2000. Shorter version given as tutorial at IEEE ISCAS-95. Support material: 300 pages.

- MATLAB for DSP Applications, This short course was given in 1997, 1998, 1999, and 2000, Support Material: 300 pages

ASU Committee Service

Department Committees

- EE Graduate Committee, Chair 1996-97, 2000-present
- Systems Area Committee, Dept. EE, Member 1988-present Chair Spring 1995, Chair 1998-present.
- Department Personnel (Promotion and tenure) Committee, member, 1993-96, 1999-2000
- Department Executive Committee, 1993-95
- Undergraduate Committee, Dept. EE, member, 1988-89, 1989-90, 1990-91, 1991-92.
- Chaired (Spring 1990) the Systems Sub-Committee (Dr. Spanias, Dr. Crouch, Dr. Grondin) of the Undergraduate Committee responsible for the review of ECE301, EEE302, EEE303, EEE405, EEE406, EEE407, EEE480, and EEE482.
- Several Faculty Search Committees, Member and Chair

College Committees

- Deans Personnel Committee, member, 2000-present
- Research Council, Member 1994-1999
- Engineering Excellence 2000 Committee, Member November 1994-95
- EE Chair Search Committee, Member 1995-96

University Committees

- Communication Advisory Committee, Member 1993-96

New Courses and Course Material Developed

- Developed a new on-line course entitled Speech Recognition ASU MEng program
-
- Developed a new on-line course entitled MATLAB for DSP Applications for the ASU MEng program

- Developed an on-line laboratory Java-DSP for EEE 407 (http://jdsp.asu.edu)

- Developed and taught a 4 Credit senior-level undergraduate course in Digital Signal Processing entitled: "Digital Signal Processing," (EEE407/591). The purpose of this course is to introduce senior students to the principles and applications of Digital Signal Processing. This course has become very popular among on-campus and off-campus students and enrollment is quite high.

- Developed and taught a graduate level special topics course entitled: "Speech Coding," (EEE 598). The purpose of this course is to introduce to graduate students the principles and applications of speech coding.

- Developed and taught a graduate level special topics course entitled: "Adaptive Filter Theory," (EEE 598 now established as EEE 606). The purpose of this course is to introduce to graduate students the principles and applications of adaptive filtering.

- Developed and supervised an advanced level independent study course entitled: "Signal Processing Using Higher Order Statistics" (EEE 790, four Ph.D. students, Spring-92). This group met once a week for three hours and material was presented from several research papers. Andreas Spanias introduced the subject during several lecture sessions and students took turns presenting the results of research papers in Higher Order Statistics.

- Developed and supervised an advanced level independent study course entitled "Speech Processing," (EEE 790, one Ph.D. student, Fall 92). The student studied voice recognition algorithms, developed software in MATLAB for one of the algorithms and presented his results in a report.

- Developed and supervised a graduate independent study course entitled "Speech Coding for Multimedia Applications," (EEE 590, one M.S. student, Spring 1994). The student studied speech coding algorithms, developed software in MATLAB for one of the algorithms and presented results in a report.

- Developed MATLAB Educational Software for Speech Coding (sample on URL http://www.eas.asu.edu/~trcsip/painter/educsw.html). Also described in a publication in the IEEE Trans. On Education ("An Educational Software Tool for the Study of Speech Coding Algorithms in a DSP Class," Andreas Spanias and Ted Painter, Special Issue on DSP Education, IEEE Trans. on Education, Vol. 39, No. 2, pp. 143-152, May 1996)

- Developed teaching material consisting of 400 viewgraphs for the DSP course (EEE 407). This was used several times in class sessions. I plan to turn this in to a book.

- Developed teaching material consisting of 250 viewgraphs for the speech coding course (EEE 598). This was made available to students along with a 100 page report developed by Andreas Spanias. It was used consistently in class sessions. This material along with software is being developed as a text book (expect completion next year).

Undergraduate Projects Supervised

- Java Software for introducing undergraduatres to DSP, Carolyn Cooper (Spring 2003). This project satisfies the requirements of the EEE 490 course.

- Array Microphones, Steve Brown (Spring 2001). This project satisfies the requirements of the EEE 490 course.

- “Real-time Spectral Analyzer on the DSP 56000,” Students: Francine Doyle and Umberto Santoni. The project entailed programming FFTs in assembly language and developing a filter bank scheme using the theory of the Discrete Fourier Transform. The software provides plots on the PC demonstrating time-varying spectral estimates. Sponsored by Motorola by donation of a DSP board. This project satisfied the requirements of the EEE 490 course.

- “Analysis Compression and Synthesis of a Speech Signal,” Student: Anastasios Policarpou and K. Gharib. The project entailed development of assembly code for the short-term linear prediction algorithm embedded in the GSM cellular standard. The project was sponsored in part by Intel (donation of the EP evaluation board). Portions of this work have integrated in a speech coder developed later for a funded project. This project satisfied the requirements of the EEE 490 course.

- “ Comparative Study of Filter Design Algorithms,” Student: J. Snider, The project entailed developing and comparing several digital filter design algorithms. Graphical demonstrations were also given. This project satisfied the requirements of the EEE 490 course.

- “Programmable Signal Generator,” Student: Bruce Negley,   This project entailed a real-time hardware implementation of programmable signal generator. This project satisfied the requirements of the EEE 490 course.

- “Analog Television Standard Conversion: A Digital Processing Approach,” Student: Brian Crawford, This projects entailed real-time hardware implementation of a digital converter PAL to NTSC. This project satisfied the requirements of the EEE 490 course.

- “Adaptive Signal Processing Algorithms,” James Chan, This project entailed developing software for a real-time visual demonstration program on the PC of the adaptation process associated with LMS algorithms. The demo program shows in color the adaptation of the poles of a second order adaptive system to the poles of a time-varying system. This software is being used in some of my classes for demonstration of an adaptive process. This was part of the EEE 490 senior project course.

- “MATLAB Implementation of the IS-54 speech coding algorithm,” Student: Renos Ioannou, The project involved development of software for the Vector Sum Excited Linear Prediction Algorithm. Sponsor: Intel (the student was paid from one of my Intel contracts).

- “ Internet-based Education,” This project started in 1996 and is being funded in part by the CIEE FGIA program.